ZX COMPUTING MONTHLY

AN ARGUS SPECIALIST PUBLICATION April 1987 £1.50

FOR ALL SINCLAIR USERS

- THE ESSENTIAL ZX CONTACTS DIRECTORY
- RANDOM LANDSCAPE GRAPHICS

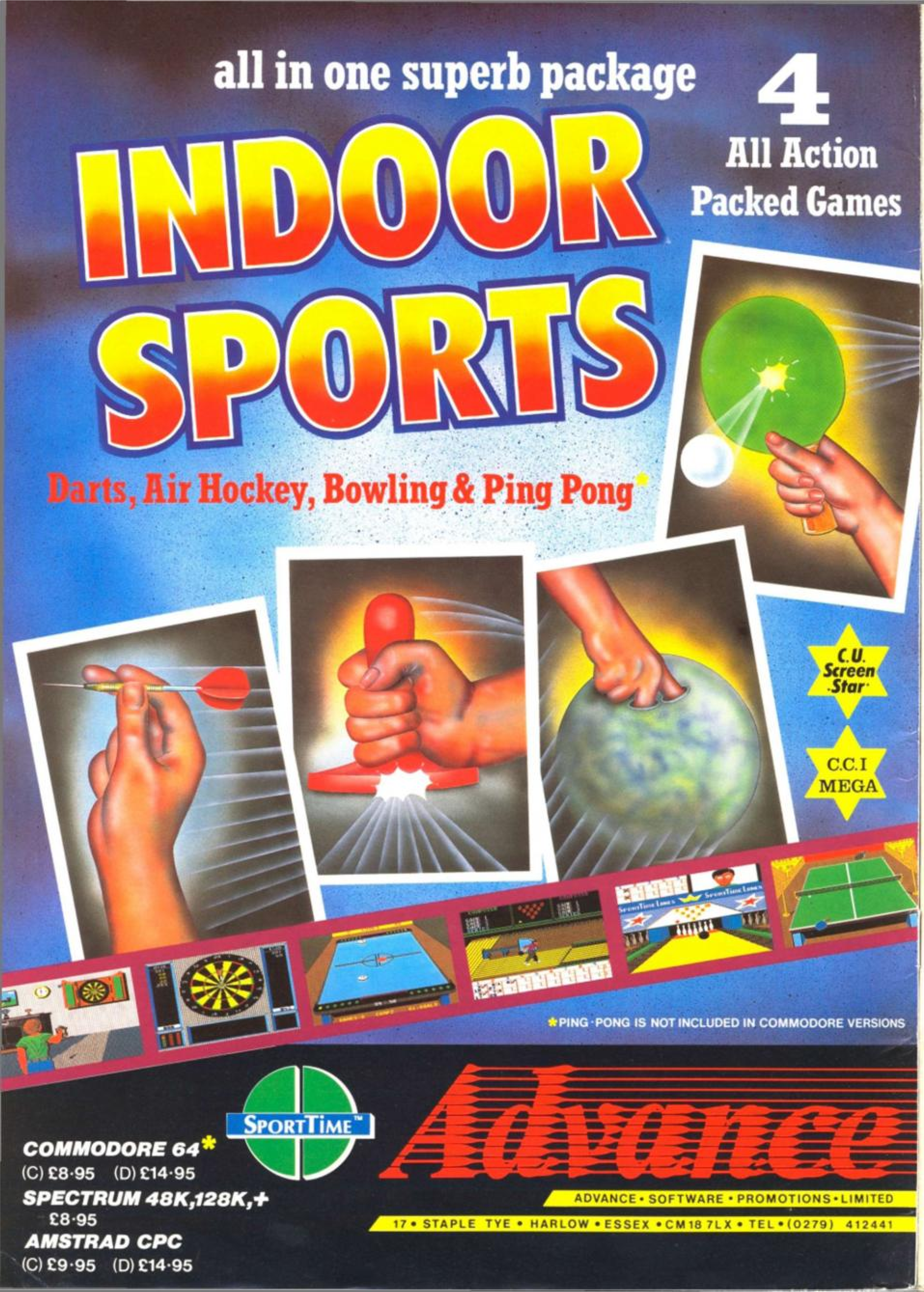
WIN THE DISCIPLE



Spectrum comic strip utility

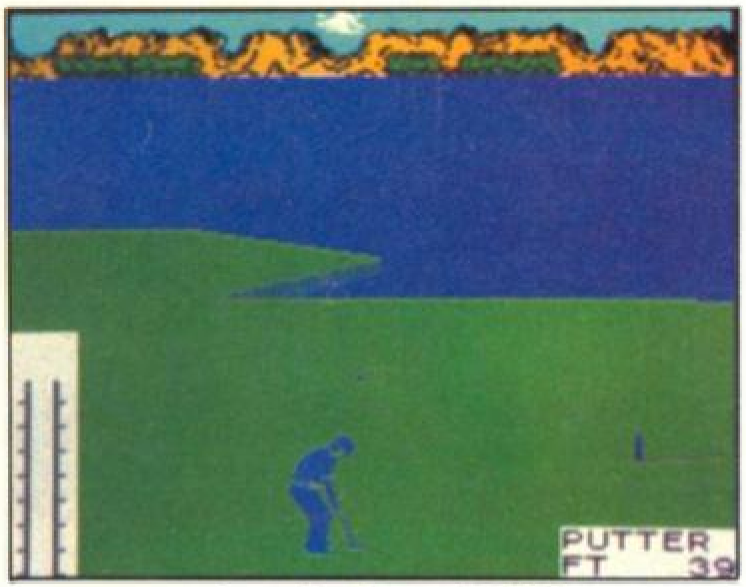
READY OR NOT... HERE IT COMES

- MONSTER HITS:
- RANARAMA
- ACE OF ACES
- LEADERBOARD



all in one superb package
4
All Action
Packed Games
INDOOR SPORTS
Darts, Air Hockey, Bowling & Ping Pong*
C.U. Screen Star
C.C.I MEGA
*PING-PONG IS NOT INCLUDED IN COMMODORE VERSIONS
SPORTTIME™
COMMODORE 64*
(C) £8·95 (D) £14·95
SPECTRUM 48K,128K,+
£8·95
AMSTRAD CPC
(C) £9·95 (D) £14·95
Advance
ADVANCE • SOFTWARE • PROMOTIONS • LIMITED
17 • STAPLE TYE • HARLOW • ESSEX • CM18 7LX • TEL • (0279) 412441

CONTENTS

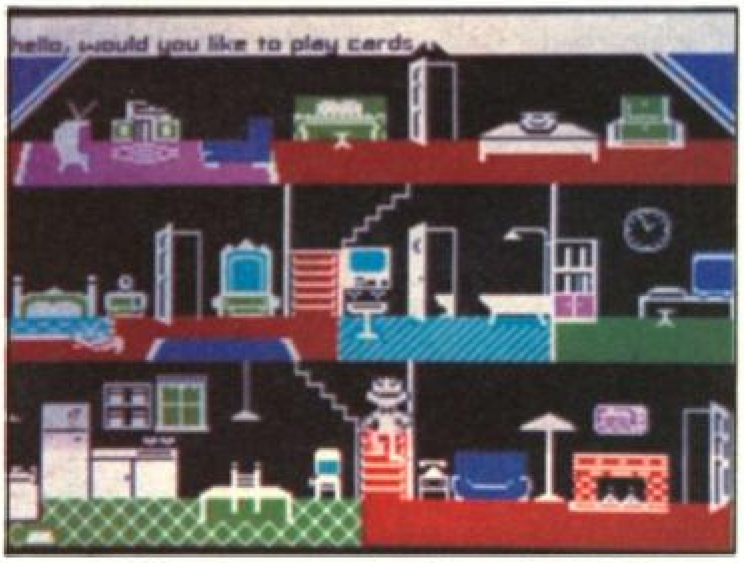


Leaderboard (48)

FEATURES

PROGRAMMING PROJECTS

REGULARS



Little Computer People (45)

SOFTWARE REVIEWS

N·E·W·S

Gremlin '87

We're being swamped by news of all the planned releases from Gremlin; as well as the reappearance of Monty Mole, Jack The Nipper, and Thing on a Spring, Gremlin have also plunged into the licensing game and are working on games based on Death Wish III (which should be easy to program, since they won't have to animate the Charles Bronson sprite), Basil The Great Mouse Detective, and the comic strip character, Mask.

Also about to appear is Krakout, a sort of cross between Space Invaders and Breakout, then there's the Samurai Trilogy (yet another martial arts game) and Star Games, a compilation tape that puts together Way of The Tiger, Beach Head 2, Barry McGuigan's Boxing and Rescue on Fractalus. These last three titles will all be available for £9.95.

Melbourne Mystery

Undeterred by the fact that they've just been gobbled up by Mastertronic, Melbourne House are carrying on as usual and are about to release The Mystery of Arkham Manor. The game casts you as an intrepid newshound attempting to uncover the unearthly mystery that surrounds the sleepy village of Arkham.

The literary trendies amongst you will of course remember that Arkham House was the company that published the stories of cosmic horror weirdy, H.P. Lovecraft, so you might well find yourself bumping into monster with lots of tentacles, eyeballs and names like Shegbag the Greebly. In addition to the eyeballs the game also features animated graphics, text, icons and pull-down menus and the adventure is so complex that it's had to be split into two parts. After a few disappointing releases it sounds as if Melbourne House may be back on form, but you'll have to wait until May before shelling out your £8.95 to find out.

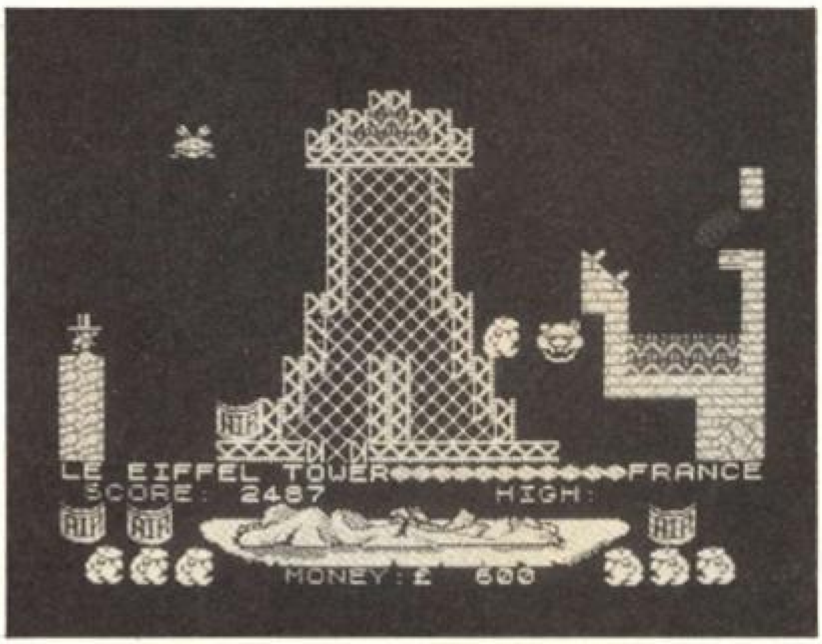
Colour of Magic Winners

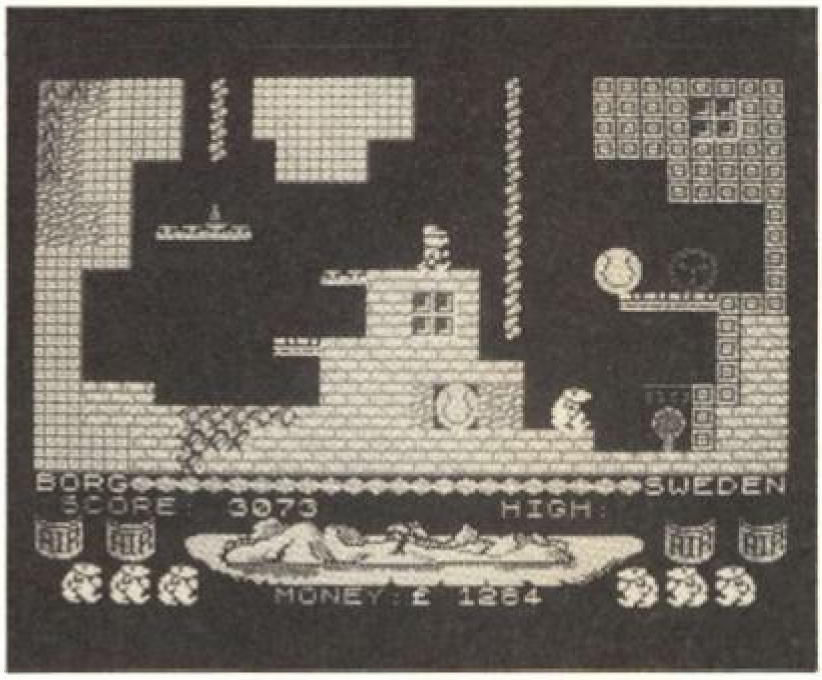
Thirty ZX readers carry off Piranha's spool adventure The Colour of Magic.

They are, Joanne Sidebottom, Hyde; Paul Oliver, Barton under Needwood; N. Owen, Prestwood; P. Luckham, Manchester; K. Rose, Oldbury; A. Brown, Northampton; A. Chodorowski, Krakow, Poland; J. Talbott, Bromsgrove; P. Underhill, Aldershot; M. Hattersley, Bushey; G. Peikner, Leitha, Austria; S. Ogleby, BFPO45; D. Riley, Spalding; K. Sagner, Bellflower, California, USA; D. Emmerson, Bridgenorth; A.G. Punchard, Farnborough; C. Renders, Farnham; K. Stephens, Hayes; H. Williamson, Moray; E. McIlwaine, Coole; T. Hoodless, Bedfore; H. Neilson, Edinburgh; M. Watson, Darwen; K. Solomon, Newcastle; C. Clennell, Blackpool; G. Deacon, Amersham; R. Verzlik, Amersfoort, Holland; K. Robertson, Harare, Zimbabwe; R. Douglaas, Mossfellssveit, Iceland; P. Lauf, Bad Vilbel, West Germany.

Disciple

In our review of the Disciple last month we gave an incorrect price. The Disciple costs £89.95 inc VAT.





The return of Monty Mole in Auf Wiedersehn Monty.

Editor: **Bryan Ralph**
Assistant Editor: **Cliff Joseph**
Consultant Editor: **Ray Elder**
Advertising Manager: **Peter Chandler**

Design: **A.S.P. Design Studio**
A.S.P. Advertising and Editorial
No. 1 Golden Square, London W1R 3AB 01-437-0626

Printed by Chase Web, Estover, Plymouth.
Advertisement Copy Controler: Andy Selwood
Distributed by: Argus Press Sales and Distribution Ltd, 12-18 Paul Street, London EC2A 4JS

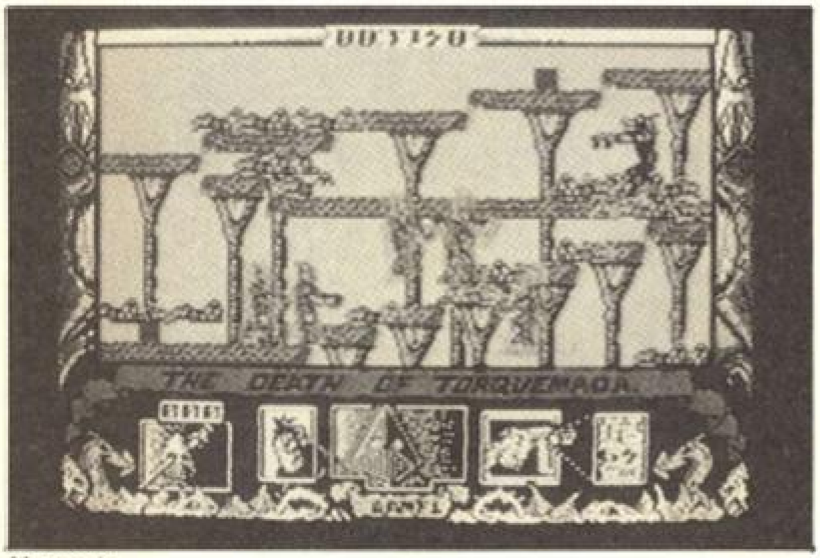
ZX Computing Monthly is published on the fourth Friday of each month. Subscription rates cn be obtained from ZX Subscriptions, Infonet, Times House, 179 The Marlowes, Hemel Hempstead, Herts HP51 1BB.

Power House

CRL's Alpha-Omega budget label hasn't exactly established itself at the front of the budget market, which probably explains why it's being relaunched as 'The Power House'. Starting off with Slingshot, by Steve Cargill (author of Melbourne House's Fighting Warrior), the new label aims to produce 'top quality games, with superb artwork at pocket money prices'. Now where have I heard that before?

CRL themselves, having brought Dracula back from the grave to the computer screen, are now busy assembling Frankenstein, a three part adventure with digitised graphics, price £9.95.

In the arcade department CRL have Sunstar due out soon, price £7.95.

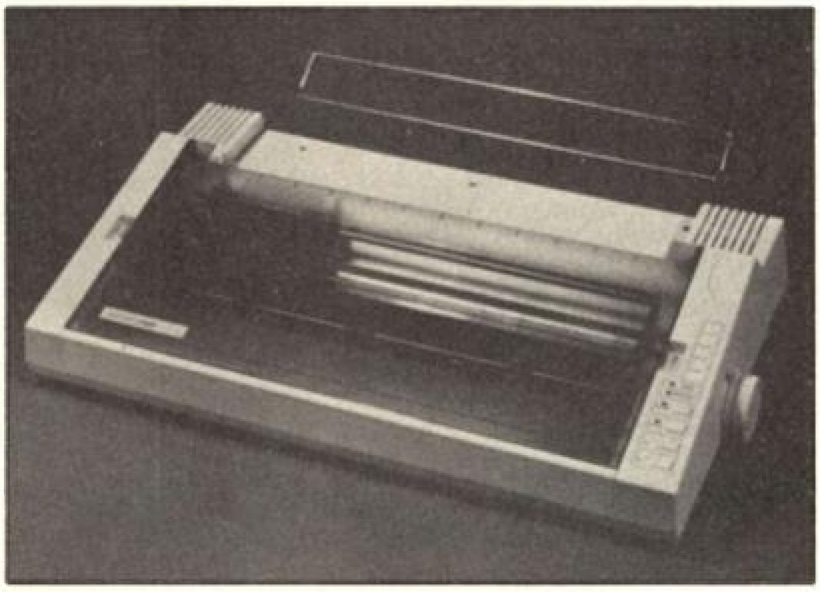


Nemesis

Throb!

Pulsator is the next release planned by Martech, in which you have to rescue a number of 'pulsies' from mazes infested with deadly alien creatures. Due for a spring release, Pulsator should also be accompanied by Nemesis the Warlock, based on the character of the same name from 2000 AD. Nemesis has to take on his arch-enemy, Torquemada, and attempt to foil his plan to destroy all life in the galaxy. Apart from their letters page there can't be much of 2000 AD that hasn't been turned into a computer game.



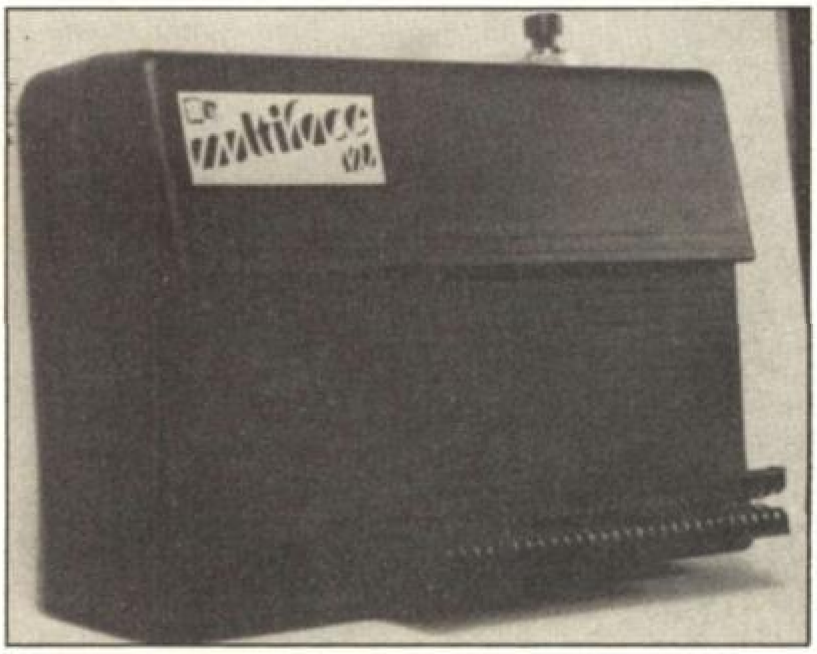


Print and be damned

Yes, it's hardware time! Following on the success of the DMP 2000 (reviewed next month), Amstrad have gone and released the DMP 4000. Positively busting with features, like a speed of 200 cps, NLQ printing and over a hundred typeface combinations, the DMP 4000 will set you back about £400.

The Cruel Sea

Activision are hoping the afterglow of Americas Cup fever will draw people to their sailing game. Due out about now, and priced at £7.99, Sailing is 'a game of strategy and simulation designed to capture the thrill of competing in top-level sailing competition'.



New Multiface

Romantic Robot have just released the follow up to the Multiface One. This new multi-purpose interface has been dubbed the Multiface 128 but despite the name it is compatible with all existing Spectrum versions. Improvements on the original include the ability to save to tape at two speeds and a function to format microdrive cartridges to over 100K.

Multiface 128 works with tape microdrive and Discovery. Versions are being prepared for the Disciple and tentatively for Beta. The Multiface 1 will retail at £44.95. Further information from Romantic Robot on 01-200-8870.

Hardball Winners

Five readers will doubtless be found in their local parks putting their new baseball kits to work. The winners of our Advance Software Hardball Competition are F. Bond, Carrickfergus, Co Antrim; A. Gairdner, Leamington Spa; Anthony Smith, Little Neston, South Wirral; French, New Milton, Hants; D. Haffner, Redditch, Worcs.

Ultimate Return

The arrival of a new Ultimate game doesn't, sadly, create the excitement that it used to, but their games are still worth looking at. April should see the release of Martianoids, in which you have to protect a vast computer, known as The Brain of Markon from the aliens who are attacking it on board its space craft. Released in conjunction with U.S. Gold, Martianoids will cost £8.99.



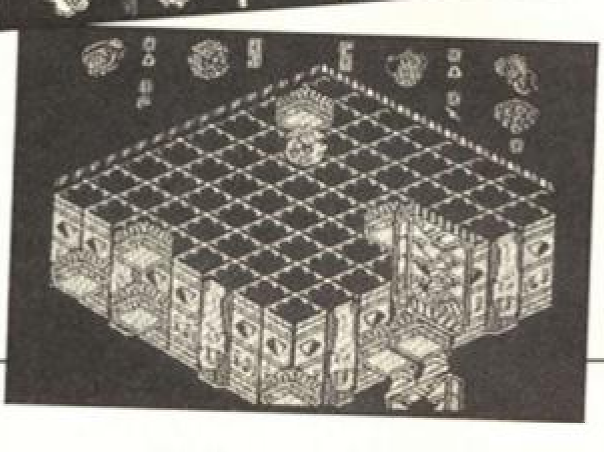
Nosferatu Winners

Piranha's adventure among the undead, Nosferatu is on its way to thirty ZX readers. They are: L. Higgins, Nottingham; M. Nicholson, Plymouth; R. Finch, Southampton; S. Goodman, Middlesbrough; G. Irvine, Co Antrim; P. Vince, London W4; N. Brownlee, Galashiels; S. Yahn, Dudley; L. Hawker, Newcastle; C. Hill, Wakefield; R. Butt, Lahore, Pakistan; Lcpl Wagland, BFPO 20; M. Page, Bridgham; J. Knight, Birmingham; M. Radley, Fife; A. James, Llanelli; D. Thompson, Runcorn; T. Woodward, Wigan; A. Lafeine, London N4; M. Suter, Bristol; M. O'Connor, Chesterfield; A. Motin, Leeds; I. McVicar, Clydebank; E. Chun, Galgate; M. Hensel, Neviges, West Germany; N. Owen, Gt Missenden; C. Thomas, Mablethorpe; J. Rimmer, Lliverpool; A. Saez, Madrid, Spain; S. Burnett, London N17.

Head Over Heels with Ocean

Here's a couple of screen shots from Head Over Heels, a new title from Ocean. Looking a little bit like an Egyptian version of Knight Lore, HoH will be available this month for £7.95. Pharaoh 'nuff?





Spectrum Games Top Ten

1	(1)	Gauntlet	U.S. Gold
2	(2)	Footballer of The Year	Gremlin
3	( )	Fist II	Melbourne House
4	(5)	Super Soccer	Ocean
5	(8)	Paperboy	Elite
6	( )	Konami's Golf	Konami
7	( )	Ace of Aces	U.S. Gold
8	( )	Konami's Coin-Op Hits	Konami
9	(6)	The Great Escape	Ocean
10	(4)	Space Harrier	Elite

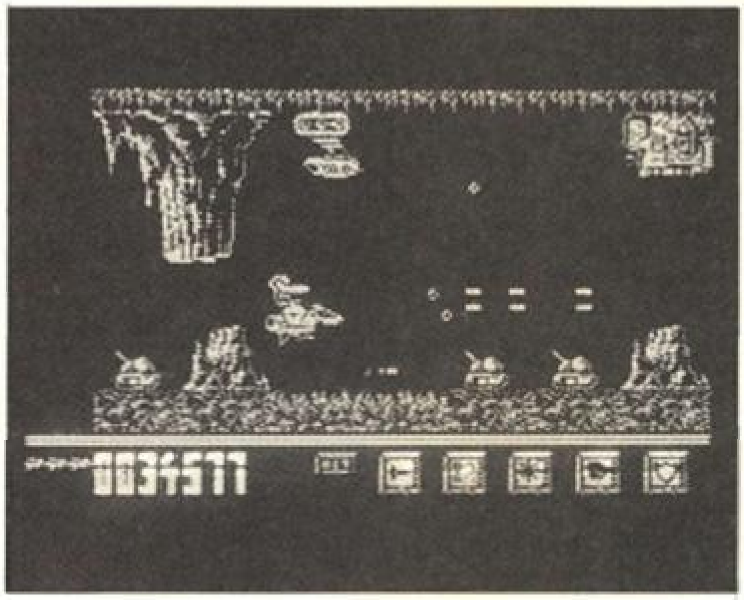
(Chart supplied by W.H. Smiths)

Saga Compliment Winner

We received a very large number of entires for this competition and everyone put forward valid reasons for why Saga's Compliment system should arrive on their doorstep. After careful deliberation we are awarding the Compliment system to Mr R. Scott of Stroud, Gloucestershire, a head master of a large primary school who sees the Compliment system as invaluable to "pioneer the unexplored opportunities for using computers to help my administration." We hope that computerisation via the Compliment assists Mr Scott and look forward to hearing about his progress with the system.

Codemasters' Brainache

Brainache, an arcade adventure, is the latest release on the Codemasters budget label, due to be followed by Transmuters, a straight arcade game featuring a combination of solid and vector graphics. Both games should be in the shops as you read this, priced £1.99.



Gremlin Clock Winners

Five Anagram champions carry off Gremlin Clocks. Ingenious displays of letter juggling came from M. Summers, Morpeth; I. Rogers, Devizes; R. Wilkinson, Rugby; H. Williamson, Elgin; A. Spall, London SE7. Among the best entries were variations on Monty on the Run, such as Hunt Torn Money and Men on Tory Hunt.

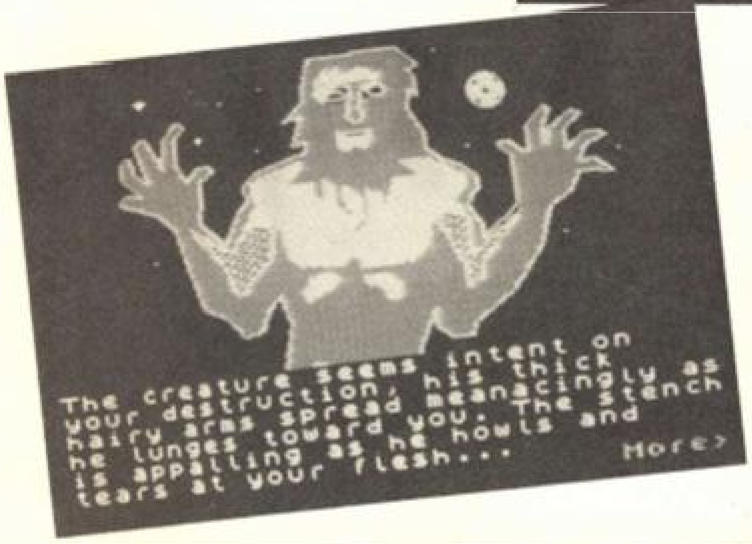
Ten runners up are R. Thrower, London; B. Carter, London E2; P.C.B. Page, Liverpool; R. Laidlaw, Birmingham; D. Garbutt, Leeds; B. Taylor, Portsmouth; S. Losch, Essen, W. Germany; S. Wright, Ipswich; D. Morrison, Aberchirder and C. Fry, Coulsdon, who will receive a copy of Trailblazer.

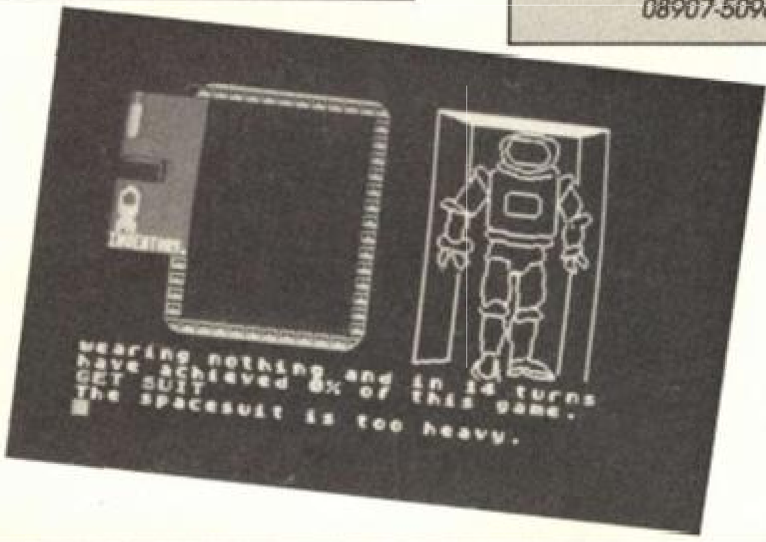


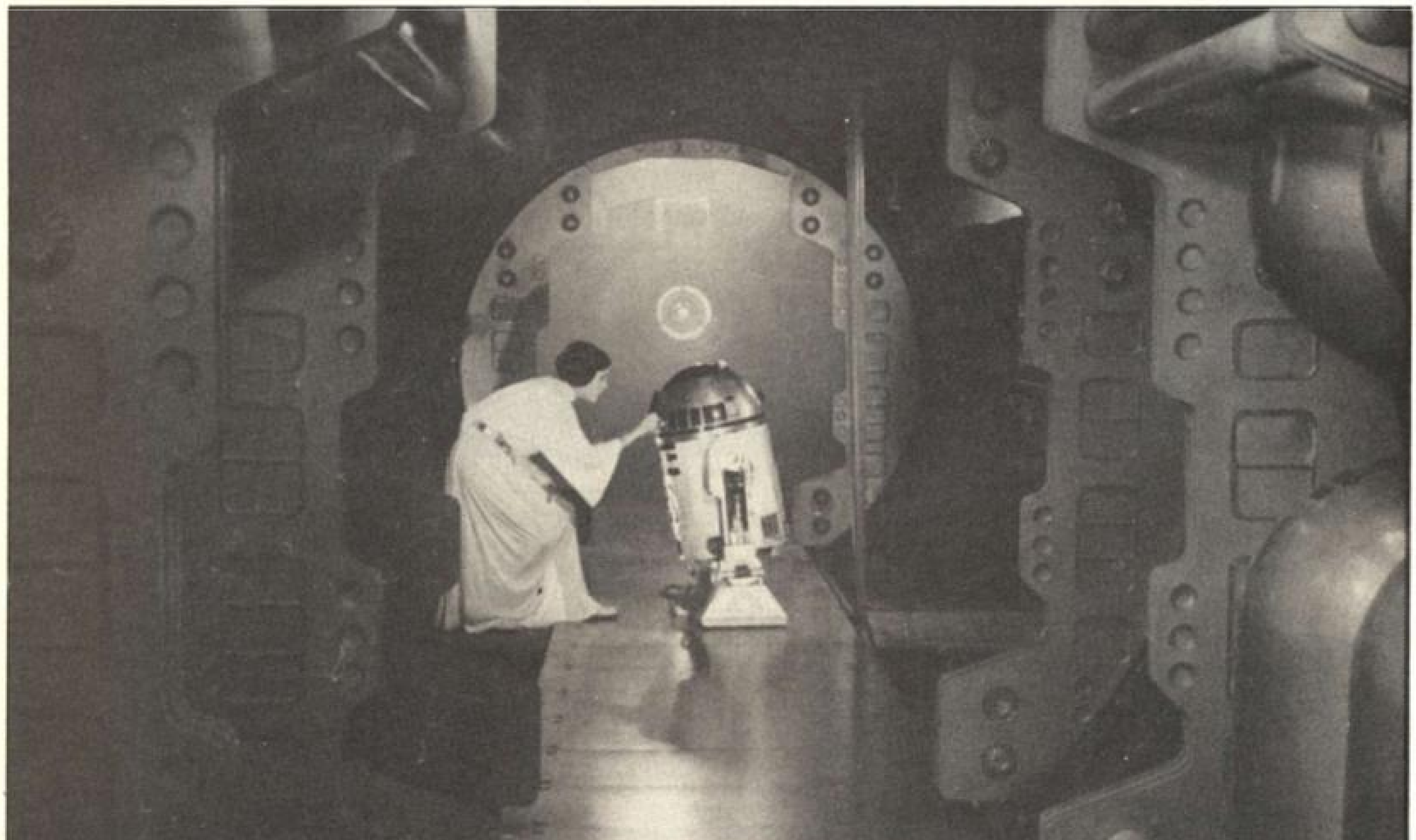
Screenshots showing the graphic capabilities of Gilsoft's Professional Adventure Writer which is nearing completion as this issue went to press.

Useful info dept.

Buying new printer ribbons all the time can get a bit pricey, but if you get in touch with Nick Godwin of Aladdink you can get your existing ribbons re-inked more cheaply than having to buy a new one. Aladdink can arrange to have ribbons of several different colours re-inked, and also offer advice about suppliers etc. You can contact Aladdink on 08907-50965.







Obi-when? kenobi

Long, long ago, in a galaxy far, far away — Da-Da! Roll credits, cue Darth Vader (wheeze, gasp, hiss) — "Your powers are weak old man . . ."

All this is just my way of letting you know that Domark have gotten hold of the rights to convert the series of Star Wars coin-op games for home computers. Lord knows when they'll appear, since they're not due for release until late '87 and programming hasn't even begun yet, so you'll just have to wait.

"You haven't learnt much young Skywalker, but you are not a Jedi yet . . ." (wheeze, gasp, hiss).

ENTER THE Z88



One year after the Amstrad buy-out, Clive Sinclair re-emerged with a new computer at the Which? Computer Show in Birmingham.

Being primarily a showcase for business computers the Which? show isn't the sort of event that would get a lot of coverage in ZX, but this year's show would include a Press conference at which Sir Clive Sinclair would announce his re-entry into the computer industry with the launch of the Z88 portable computer. So, notepad in hand, I headed north to witness the Second Coming.

After taking a wrong turn and finding myself in the Caravan and Boating Exhibition (that wind-powered disc drive turned out to be a 20ft yacht) I eventually found my way into Halls 4 and 5 of Birmingham's National Exhibition Centre.

As you'd expect at a business exhibition the hardware and software on show tended to be functional rather than entertaining. Everywhere you looked there were printers, from relatively cheap (under £500) dot-matrix jobs right up to top of the range laser printers that would be out of reach of all but the wealthiest home computer owners. Citizen were there with, among others, their LSP-10 and 120D models which were probably the cheapest machines on show at £275 and £189 each. Both of these got the thumbs-up in recent issues of ZX, and it seems that a lot of people like Citizen printers because the company claim to be selling a printer somewhere in the world every four minutes.

Printers apart, the main thrust of the show seemed to be into the field of desktop publishing. Amstrad, Commodore, Apple, Apricot, all the big names were there with desktop packages aimed at turning us all into mini Robert Maxwells. That sort of technology is slowly coming within reach of home computer owners (Softek's Artist II and Writer offer 'pagemaker' facilities of a sort), but I think we'll have to wait for the next generation of Spectrums before any daily newspapers start rolling out of our front rooms.

The closest that the various

exhibitors got to home computing was with a number of Amiga and Atari ST packages that were a bit boring anyway, although Psion's Chess is worth watching on any machine. Tatung were there plugging the Einstein 526, but nobody seemed to be paying much attention.

Creative Sparks Distribution were there, announcing their new Status label. The first releases on the label are Quest For The Ring and Fortune Teller, two games for that most popular of home computers, the IBM, but a few Status games should eventually find their way onto the Spectrum.

The high spot of the morning though, was an advertising video produced by one company in which John Cleese attempts to run IBM software on a dead fish. Well, it was better than watching another load of printers churning out sales figures.

Come 1:30 it was off to the Press office for Sir Clive's great announcement. The new machine, it turned out, is called the Z88 and is produced by Cambridge Computer Ltd, a new company partly owned by Sinclair Research and with Sir Clive as chairman.

Lightweight division

We all knew that it was intended to be a portable business machine, since that had been announced even before last year's deal with Amstrad. Previous attempts at providing a decent amount of computing power in a portable machine had, according to Sir Clive, resulted in compromises in weight resulting in machines weighing in at about a full stone. The Z88 is smaller than a sheet of A4 paper (in other words it would fit onto this page), weighs under two pounds and will cost about £230 including VAT.

There were no machines available for bench testing, but the specs sounded interesting. It has a basic 32K RAM which can be expanded up to 3Mbytes with 32, 128 and 1Mbyte cartridges. The 128K ROM holds the operating system, wordprocessing and spreadsheet software, along with a few extra bits and pieces like a calculator, diary, and a clock/calendar.

The Z88 runs BBC Basic "because it's fast and widely known". The BBC machines are 6502 based while the Z88 is still Z80 based, but is comparable to the BBC's second processor according to Sir Clive, though BBC programs couldn't run on the machine because of differences in the display.

In addition, while the Z88 won't actually run IBM software (what do you expect for 200?) it does allow you to transfer data files, such as spreadsheets or wordprocessing files to and from IBM compatible machines. There is a built-in RS232 port, and a modem (£100) will soon be available.

The whole shebang runs on four penlight batteries, which give 20 hours of 'computing time'. This means 20 hours of actually using the machine for processing information, but when the machine is not actually in use the batteries will keep the contents of memory intact for about a year so you shouldn't have to worry about losing vital information when the batteries pack up (or for that matter when you take the batteries out to replace them, since there is a back-up which protects the memory while this is done).

The display is a form of LCD, called 'Supertwist' — which means that you don't have to stand upside down and look at the display from the right angle in order to see it clearly. The layout consists of a central display of eight lines of 80 characters, and on each side of this there is a window. The left hand window is a menu containing the various operating modes of the Z88, while on the right is an interesting feature — a mini-display of the full 64-line page which is updated as work is carried out. This is the sort of feature that is only available on some of the most expensive wordpro machines, so getting it onto a portable is quite clever.

After the Press conference I asked Sir Clive what happened to Sinclair Research's own flat screen technology. Had that ever been considered?

"We looked at it for a long time," he told me. "But we couldn't get 80 characters onto it, and everyone told us they wanted 80."

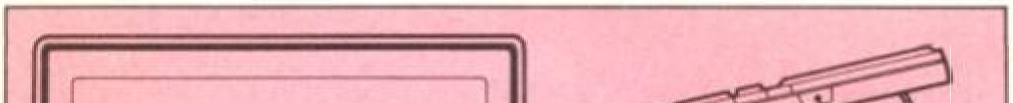
If the Z88 lives up to its aims then Cambridge Computer could have a success on their hands. The specifications are as good as you're likely to get from a portable machine and much cheaper than any of the competition. I doubt if it'll set the world alight the way the Spectrum once did or fail as spectacularly as the QL. On the whole it seems to be a fairly sound machine, sensibly aimed at a strong market and should provide Sir Clive with a much needed boost to his commercial credibility.

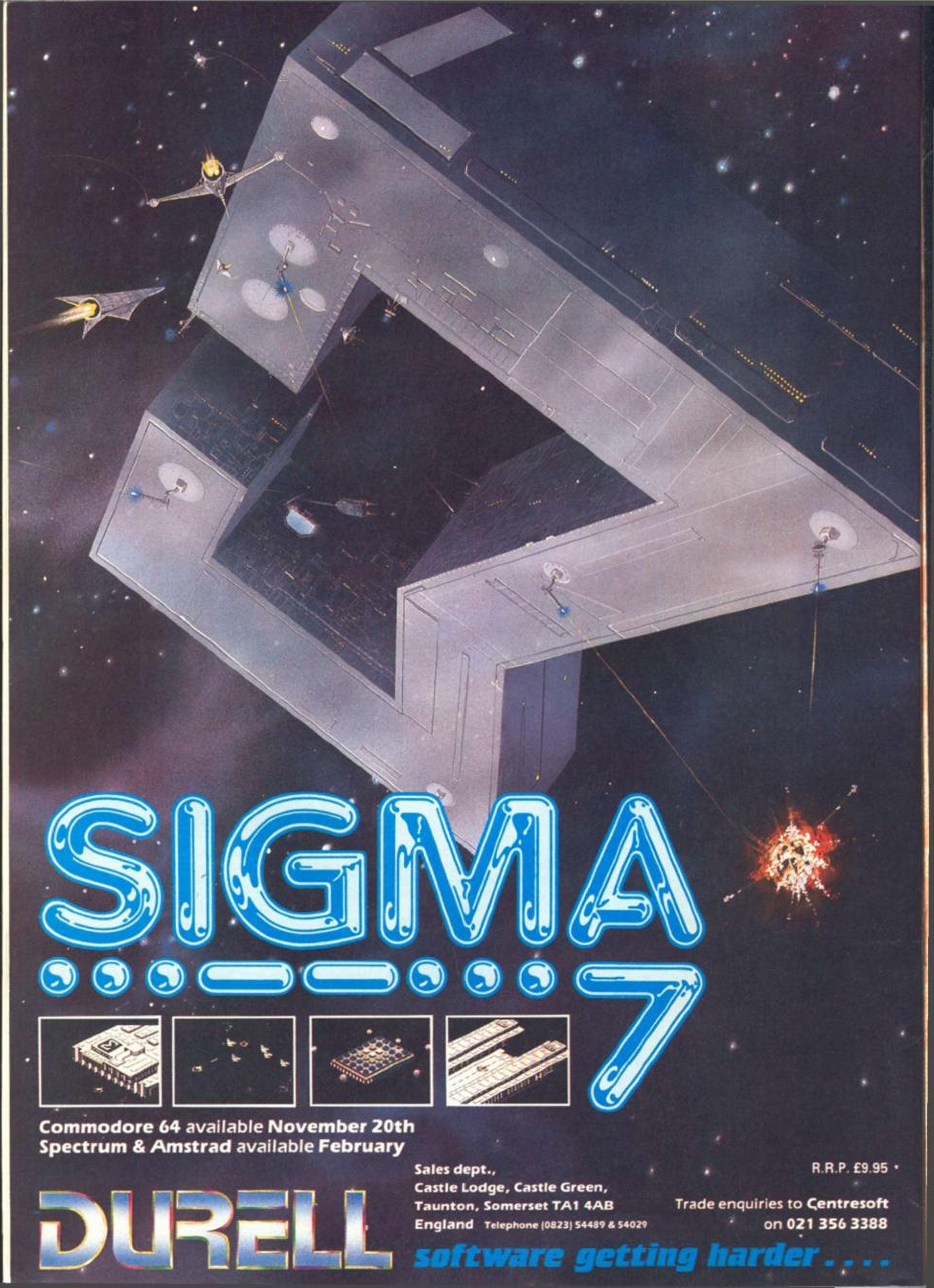
Silent keyboard

The only slightly dubious feature of the machine is that old Sinclair favourite, the keyboard. They wanted a keyboard which would be "virtually silent" so that it could be used in meetings without rattling away and getting on everyone's nerves. So they've opted for a keyboard moulded out of 'silicon rubber'. The whole thing is moulded out of a single sheet of this material, and while it's claimed to be good enough for touch typing it does bring back memories of the old rubbery Spectrum keyboard. That's a feature that a lot of people will probably be watching quite closely, though in the brief time that I got my hands on one it did seem to be better than the Spectrum board.

The machine is due for an April launch and will initially be available by mail order (which prompted one person to ask "does the 88 refer to the year when it will arrive"?). It will be aimed mainly at the business market, though Sir Clive added that there might be an educational use. It was priced, he said "so as not to give anyone a reason for not buying it." So does he see the Z88 creating its own market, as the ZX81 and Spectrum did, or does he simply hope to fill a gap in the already established business market?

"Well, you always aim at a market and we do have businesses and domestic use in mind, but you do get surprised. For instance the Spectrum went very games orientated." Which is about where I came in.




SIGMA 7
Commodore 64 available November 20th
Spectrum & Amstrad available February
DURELL
Sales dept.,
Castle Lodge, Castle Green,
Taunton, Somerset TA1 4AB
England Telephone (0823) 54489 & 54029
R.R.P. £9.95
Trade enquiries to Centresoft
on 021 356 3388
software getting harder....

DURELL



SABOTEUR II
Avenging Angel

Available on Spectrum, Amstrad & Commodore

DURELL sales dept.,
Castle Lodge, Castle Green, Taunton, Somerset, TA1 4AB
Telephone (0823) 54489

R.R.P. £7.95

CROSSWIRES

Ray Elder tackles more technical nightmares

Chip off the old block

Q *Dear Sir,*
Please could you help me with a problem I am having with TASWORD 2. I have a Spectrum+ connected to a Centronics GLP 2 printer by a ZXprint 3 interface and when using Tasword 2 everything works perfectly except when I use block move or block copy commands.

Both of these commands work perfectly on the screen but when it comes to printing files in which they have been used all I get is a system crash. As I mentioned earlier it's only a minor problem but infuriating when I have a lot of repetitive text that has to be typed out every time. Have you or any of your readers come across this problem before? Any help would be gratefully appreciated.
Yours faithfully,
Richard Boyles

A I hardly use these functions but apart from having a LX86 printer in place of your GLP our systems are the same. As a test I used these move and copy functions several times and then tried a print, with no problems.

All I can suggest is that there may be a corrupted bit of code in your version of Tasword, if you are not using the original tape but a backup copy then try setting up a new backup copy and see if the problem persists. As a last resort send me a copy to try on my set up or try it with someone else's system and if it fails to work then your master tape has a fault.

Bright eyes

Q *Dear Mr. Elder,*
I recently upgraded to a 128K Spectrum, and I seem to be having attribute problems. Even at power up the Sinclair Logo has unwanted yellow and blue attributes, which also occur on the pull down menu.

I returned my first 128K to Comet who duly gave me a replacement. This appeared faulty as well. They obligingly tried out further 128s and plus 2 and all had the same problem. So did those in other shops locally. Could the current atmosphere conditions be causing the problem, have we swapped dot crawl for a more infuriating display? Please help.

My old Spectrum works perfectly.
Yours Sincerely
Tony North

A Atmospheric conditions is a new one on me, but who can say! I have seen and used some four/five 128s and Plus2s and none have had any discolouration problems. You do not say whether you have used the same TV for all these machines and if you have then it may be an unusual case of incompatability between the new machines and your set. If you have tried them out on different sets then it is possible that a faulty batch of machines was supplied to Rochester, but I can assure you that this does not seem to be a common problem.

Opus out of tune

Q *I have a Spectrum 128+2 and I have recently bought an Opus Discovery. I am unable to get commands into the Discovery. After five phone calls to Opus who gave me various checks to do, it still does not work.*

I borrowed a Spectrum+ and the Discovery worked a treat. On informing Opus they suggested that I enter the command ***PRINT USR 8****, which should give me 2.2 in the top left hand corner of the screen. What it gave was:*
48K mode 5 M,0:1
128K mode B(*K/,0:1
Opus assured me that it should work on both models. It's been five weeks now and I'm still not up and working. Could it be a defective I/O port on the computer? I would appreciate it if you could come up with any ideas please.
Yours faithfully,
D.F. Hackney

A You could be right with the defective I/O, but I doubt it. Opus produced a separate version of the Discovery for the 128, which essentially had a different ROM. The fact that it worked on a Plus and that it didn't produce the 2.2 result seems to indicate that it is the old, 48K only, ROM in the Discovery.

Unless you bought it secondhand I would return it to the shop and take along your computer so they can see for themselves that it won't work. If you did buy it secondhand then your only recourse may be to contact Opus yet again and either get them to look at it for you (unlikely in view of their pulling out of the market) or supplying a 128+2 ROM. Both of which might cost you some cash.

QL ERROR error?

Q *Dear Sir,*
I'm afraid your QL error trapping routines won't work on mine! I tried them out but got the message "Not implemented at line. . . ." I hope this does not mean I have got some kind of obsolete ROM as my QL was bought only recently from Dixons.
Yours sincerely,
Mr. C. Hopkin

A Actually ALL QLs are now technically obsolete! Seriously though there were several versions of the ROM, each having slightly different features. Unfortunately we do not have machines with all versions of the ROMs at our disposal so cannot speak for all of them. The one we used accepted the error trapping routines. As this was a non official routine (Sinclair did not claim it in the specs or acknowledge it in the manual) I'm afraid you will simply have to accept that it is not available to you. Sorry.

DISKussion

Q *Dear Sir,*
At the present moment I own a Spectrum+, Multiface 1, Rotronics Wafadrive & Timex 2040 printer. Recently I discovered that Rotronics had gone into liquidation which prompted me to consider a new form of storage system. Due to some program incompatibility and the difficulty of using the operating commands I have been thinking of changing for some time. I decided the Opus system would suit me but now I hear that the range is being discontinued. The only option seems to be to buy a seperate disk drive interface and disk drive, but I know relatively little about either except that the one which seems to offer the most is the Disciple.

Could you please tell me which of the various interfaces and disk drives would be cheapest, most compatible with Spectrum commands and Multiface 1, and meet my storage needs.

Finally, as I do not know whether any disk drive and interface combination would be compatible with the Timex printer, it would seem a Centronics printer and interface may be needed. The Kempston E interface has been highly recommended and I was thinking of the Brother HR5 printer.
Yours faithfully
K.J. Bryan

A It sounds as though your needs mirror my own to a great extent, except I probably do a lot more wordprocessing, and I have been using the Technology Research Ltd. system for the past three years (the same as the Cumana system). The Beta+ interface has its own built-in multiface type "save program" option so you can sell your Multiface to offset the cost. It does work with the TRL though. The TRL also allows you to select the type of disk drive you want to use, it will support up to four 5.25 3.5 or 3" units or a combination of them. I use a 5.25, but would probably go for the 3.5 if buying nowadays.

The operating syntax is quite different to Sinclair's but once learnt is easy to remember. I converted and run TAS-

WORD 2, MASTERFILE and use the SAVE option to store several programs, although not all are compatible (around 80% are OK). The printer/interface combination sounds good, they are compatible with the TRL, but then so is the 2040, it's worth the extra though if you do any serious writing. I use a Euro-electronics ZXLprint 3 and that also is a compatible device. Nothing will ever provide a perfect system; mine suits me, but may not be your cup of tea. Try and get demonstrations of the units you are interested in before buying and then you'll have some idea of what it can do. To try and discuss the whole range still available would take several pages of an article!

Interface

Q *Dear Mr. Elder*

I would appreciate your advice on interfacing a Spectrum+2 with a Star SG-10 printer. The printer has been used by a friend who has a BBC and used it with View. I have just purchased Taswórd 2 and find it a reasonably friendly wordprocessor. I have just purchased Tasword 2 and find it a reasonably friendly wordprocessor.

Having obtained the printer, my first idea was to purchase an interface from Tasman but, having read Alan Davis' article in the January issue, I am now not so sure. Would a Kempston E type interface be better? I find the whole idea of interfacing very difficult. I am please with the performance of the Plus 2 and find it a versatile machine.

Thanking you in advance.

Yours sincerely,

David Muir

A The idea of interfacing is actually quite simple, an interface simply allows instructions to be sent from one device to another in a form that it can understand. There are two main printer interfaces, Centronics which is a parallel signal sending device, transferring data in whole bytes (eight bits), and RS232 which is serial, transferring data one bit at a time.

As your printer was ex-BBC it is almost certainly Centronics and so a suitable interface is required. The Tasman one is very good provided you want to use it primarily with TASWORD, there are problems if you want to print any graphics, and separate software is needed. Both the Kempston E and the ZXLprint 3 interfaces have built-in software and can cope with nearly every printing situation you are likely to encounter. Both are reasonably easy to use and are highly recommended.

Spectramusical

Q *Dear Mr. Elder,*

I read with great interest your review of the RAM ELECTRONICS MUSIC MACHINE and being interested in home recording I thought this was just what I'd been waiting for. Then I read the review of the Cheetah sound sampler and I am now in a quandry as to which to buy, perhaps you would answer a few questions about the units.

1. *Does the Cheetah have tune sequencer facilities like the Music Machine?*
2. *Would the Cheetah keyboard be compatible with the Music Machine or would the Cheetah MIDI interface be needed as well?*
3. *With the keyboard attached could either be played polyphonically?*

I would be grateful for your answers and if you would tell me which is the better buy.

Yours faithfully,

Chris Avery

A First of all it is only fair to say that personal preference has a lot to do with the choice and I stress that I am offering my own opinion and that others may not agree with me.

1. No
2. The Cheetah keyboard will connect directly to the Music Machine without further interfaces, via the Machine's MIDI sockets.
3. No

In my opinion the Music Machine is a much more versatile unit and the Cheetah keyboard is excellent value for money, however I would suggest that you consider pruchasing a real synth if you want polyphonic capability and the CASIO CZ101 is a marvellous machine which can be interfaced to your computer by any MIDI unit including the Music Machine. It has very sophisticated MIDI implementation. You can buy a CZ new for around two hundred and fifty pounds or for much less secondhand.



PROJECT X

Ray Elder presents readers' solutions to the first Project X programming challenge.

When I wrote the first of these articles neither I nor the Editor had any idea of the volume of response we would receive. In fact there were so many replies that it has taken me until now to read them all and write this follow-up article.

To refresh your memories, the project was to write a program to decode the kind of message which has been encoded by shifting the alphabet.

We had hundreds of solutions, all of which were valid, and the general standard of programming was so high that picking just one winner proved impossible. So my solution is to print a selection of the best and to mention those readers whose letters or programs I particularly noted. Please do not feel offended if I missed you out!

The solutions were essentially based on one of three ideas, a set of loops shifting the alphabet, resetting the 'chars' pointer and using the FN function. In between we had variations in style, user friendliness and even a reasonably successful attempt at artificial intelligence.

Project leaders

Camiel Devos lives in Belgium, has Hercule Poirot as a hero, and provided the program with some AI. His idea was to check each word decoded for impossible letter combinations and skip to the next shift if valid. The rule he uses is that there are never more than four consonants together and short words contain at least one vowel.

Ricky Han used the FN to help him produce a three line solution, **Chris Oliver** also used the FN to produce a coder and decoder, however, both his programs were identical except for the messages, perhaps the two could be combined, a code/decode option given and the messages presented depending on the option selected?

Tom Moore used the 'chars' pointer method in 14 lines, **Kevin Oscroft** took 9 lines and **Charles Webb** did it in 5 lines. Not that length matters (this isn't shortcuts!).

Talking of length **Ron Cavers,** of Chezron Software, sent in a one line solution, he also sent in an expanded version which was easier to understand. Even that could have been shortened by changing the third statement of the line to LET B$=A$.

Mr. R. J. Woodham created both coder and decoder programs controlled by menu selection and sent a pleasant message — coded of course — **Rene Uittenbogaard** included a printer option, **Lee Corbin** turned all the letters into their codes and manipulated them in a numerical array.

P. S. Robson, along with many others, used POKE 23658,8 to set CAPS LOCK on and ensure all input was in capital letters then used the conversion to numerical value method. **Norman Strong** added to the discussion by pointing out that time could be saved by looking at a short selection of a long message, perhaps 15 letters, as this would be enough to see if it was unintelligible or not.

Mr. P. A. Edwards used seven lines and suggested that I was not really trying with my own 23 line effort, and **Mark Moody** used loads of REMs as we are told good programmers should. **Mr. B. Rumbelow** used an interesting two dimensional array with the FN function.

Finally, a mention for the following readers who produced programs of equal merit but I just couldn't get them all in, and a special thanks to **John E. Amphlett** who not only sent in an interesting letter pointing out that we were really discussing CYPHERS and not CODES (a rose by any other name etc. etc.), but also put forward an idea for a future Project X topic.

Mentions to: **Steven Anthony, Ian Howie, S. M. Goodman, Sgt. Dave Brooks** (I know RAF Bruggen well from around 10 years ago!), **Ciaran Gultnieks** and **John Chamberlain.**

Small tokens of appreciation are on their way to all who got their progs published and my thanks to all who wrote in to me.

Now I must away to start looking at the programs sent in for the last Project X and begin planning Project X Mk3!

Project X solutions

```
   1 REM Camiel Devos
   5 INPUT "Words to decode: ";i
$
  10 INPUT "Places to shift 1-25
            or 0 for computer ch
oice: ";s
  15 IF s=0 THEN  LET s=INT (1+R
ND*25)
  20 FOR n=1 TO LEN i$
  25 IF i$(n)=" " THEN  NEXT n
  30 LET i$(n)=CHR$ (CODE i$(n)+
s)
  35 IF CODE i$(n)>122 THEN  LET
 i$(n)=CHR$ (CODE i$(n)-26): GO
TO 45
  40 IF CODE i$(n)>90 AND CODE i
$(n)-s<91 THEN  LET i$(n)=CHR$ (
CODE i$(n)-26)
  45 NEXT n
  50 LET e$=i$+" "
 100 FOR n=1 TO LEN e$
 110 IF e$(n)<>" " THEN  PRINT e
$(n);
 120 NEXT n
 130 PRINT AT 20,0;"Just wait. I
'll do my best.": LET s=1
 140 IF s>25 THEN  PRINT AT 20,0
;"This seems impossible to solve
  with this decoder.": PAUSE 0:
CLS : GO TO 5
 150 LET m=0: LET c=0: LET x$=e$
```

```
 160 FOR n=1 TO LEN x$
 170 LET m=m+1: LET c=c+1
 180 IF x$(n)=" " THEN  GO TO 22
0
 190 LET x$(n)=CHR$ (CODE x$(n)+
s)
 200 IF CODE x$(n)>122 THEN  LET
 x$(n)=CHR$ (CODE x$(n)-26): GO
TO 220
 210 IF CODE x$(n)>90 AND CODE x
$(n)-s<91 THEN  LET x$(n)=CHR$ (
CODE x$(n)-26)
 220 IF x$(n)="a" OR x$(n)="A" O
R x$(n)="e" OR x$(n)="E" OR x$(n
)="i" OR x$(n)="I" OR x$(n)="o"
OR x$(n)="O" OR x$(n)="u" OR x$(
n)="U" OR x$(n)="y" OR x$(n)="Y"
 THEN  LET c=0
 230 IF x$(n)=" " AND c=m OR c>4
 AND x$(n)<>" " THEN  LET s=s+1:
 GO TO 140
 240 IF x$(n)=" " THEN  LET m=0:
 LET c=0
 250 NEXT n
 260 PRINT AT 10,0;x$;AT 20,0;"I
s this alright ?               "
 270 INPUT "OK ? y/n ";o$
 280 IF o$="y" OR o$="Y" THEN  G
O TO 300
 290 PRINT AT 20,0;"Sorry, I'll
try again.          ": LET s=s+1
: GO TO 140
 300 PRINT AT 20,0;"You see,it's
 easy.               Any key to r
estart.": PAUSE 0: GO TO 5
```

```
   1 REM Ricky Han
  10 DEF FN b$(x)=CHR$ (n+x-26*(
n+x>90)): POKE 23658,8: INPUT "E
nter coded message.            "
;a$: FOR n=1 TO 26: FOR a=1 TO L
EN a$: IF a$(a)=" " THEN  PRINT
" ";: GO TO 30
  20 PRINT FN b$(CODE a$(a));
  30 NEXT a: PRINT '"Press any k
ey.": PAUSE 0: CLS : NEXT n: STO
P
```

```
   1 REM RON CAVERS
  10 INPUT "Please enter coded t
ext. ";M$: LET A$="ABCDEFGHIJKLM
NOPQRSTUVWXYZ": LET B$="ABCDEFGH
IJKLMNOPQRSTUVWXYZ": FOR H=1 TO
25: LET B$=B$(2 TO )+B$(1): FOR
F=1 TO LEN M$: FOR G=1 TO 26: PR
INT A$(G) AND M$(F)=B$(G);: NEXT
 G: NEXT F: PRINT : NEXT H
```

```
   1 REM LEE CORBIN
  10  POKE 23658,8: BORDER 0: PA
PER 0: INK 7: CLS : INPUT "YOUR
MESSAGE ? ",, LINE A$
  20 DIM Z(LEN A$): FOR B=1 TO 2
6
  30 FOR A=1 TO LEN A$
  40 IF A$(A)=" " THEN  LET Z(A)
=32: NEXT A
  50 IF CODE A$(A)-B<65 THEN  LE
T X=CODE A$(A)-65: LET Z(A)=91-(
B-X): GO TO 70
  60 LET Z(A)=CODE A$(A)-B
  70 NEXT A
  80 PRINT ,: FOR A=1 TO LEN A$:
 PRINT CHR$ Z(A);: NEXT A
  90 NEXT B
 100 PAUSE 0: GO TO 10
```

```
   1 REM MARK MOODY
  10 POKE 23658,8: REM CAPS LOCK
  20 CLS : INPUT "TYPE IN YOUR M
ESSAGE:-               ";A$
  30 LET B$=A$: LET L=LEN A$
  40 FOR F=1 TO 26: FOR N=1 TO L
  60 LET C=CODE A$(N): IF C=32 T
HEN  NEXT N: REM SPACE=CHR$ 32
  65 LET C=C+F: IF C>90 THEN  LE
T C=C-26: REM WORD WRAP
  70 LET B$(N)=CHR$ C
  80 NEXT N
  90 PRINT AT 10,0;B$
 100 BEEP .1,0: PAUSE 50: NEXT F
```

```
   1 REM TOM MOORE
  10 FOR I=0 TO 207
  20 POKE 30216+I,PEEK (15880+I)
  30 POKE 30424+I,PEEK (15880+I)
  40 NEXT I
  50 INPUT "1. INPUT TEXT IN FUL
L.             2. USE ONLY CAPITALS
 AND SPACES 3. DECODED VERSION O
F THE TEXT      IS SHOWN IN FULL,
 OFFSET BY      ONE LETTER AT A T
IME.            4. THE NEW LETTER FO
R 'A' IN THE    ORIGINAL TEXT IS
GIVEN AT THE    BOTTOM LEFT CORNE
R.              5. PRESS ENTER TO GE
T THE NEXT      VERSION OF THE TE
XT.             ";C$
  60 POKE 23607,116
  70 FOR N=25 TO 0 STEP -1
  80 CLS
  90 PRINT C$
 100 INPUT "A
            ";V$
 110 POKE 23606,N*8
 120 NEXT N
 130 POKE 23607,60
 140 STOP
```

```
   1 REM Mr.B.Rumbelow
   2 REM This program also tries
         to solve anagrams and
         even coded anagrams.
  10 POKE 23658,8: LET tot=0
  20 INPUT "Phrase is (NO SPACES
) ";A$: LET l=LEN A$: DIM z$(4,2
6)
  30 LET z$(3)(1 TO )=A$
  40 FOR m=1 TO 2: FOR n=1 TO 26
: LET z$(m)(n TO n)=CHR$ (n+64):
 NEXT n: NEXT m
  50 DEF FN f$(A$,A,N)=(A$( TO N
-1) AND N>1)+A$(A+N-1)+A$(N TO A
+N-2)+A$(A+N TO )
  60 LET N=1: DIM A(l): DIM B$(l
-1,l): LET B$(N)=A$
  70 GO SUB 90
  80 PRINT "THATS IT FOR ";A$: G
O TO 220
  90 LET A(N)=1
 100 IF l-N+1>2 THEN  LET N=N+1:
 LET B$(N)=FN f$(B$(N-1),A(N-1),
N-1): GO SUB 90: GO TO 130
 110 POKE 23692,255
 120 LET z$(3)(1 TO )=FN f$(B$(N
),A(N),N): GO SUB 160
 130 IF A(N)<l-N+1 THEN  LET A(N
)=A(N)+1: GO TO 100
 140 IF N>1 THEN  LET N=N-1
 150 RETURN
 160 PRINT  INVERSE 1;z$(3)(1 TO
 l): INVERSE 0
 170 FOR q=1 TO 26
 180 LET z$(2)(1 TO )=z$(2)(2 TO
 )+z$(2)( TO 1)
 190 FOR k=1 TO l: LET v=(CODE z
$(3)(k TO k))-64: LET z$(4)(k TO
 k)=z$(2)(v TO v): NEXT k
 200 PRINT z$(4)(1 TO l),: LET t
ot=tot+1: NEXT q
 210 RETURN
 220 PRINT '': FOR n=1 TO 4: PRI
NT z$(n)(1 TO ): NEXT n: PRINT
tot;" Combinations": STOP
```

DISCIPLE DATA

John Wase documents some new commands for the new interface.

A relative newcomer on the scene, the Disciple, as you probably know is a disc interface/magic box. One of its features is that part of the software loads in from disc on booting up the system: don't worry, it doesn't corrupt your program but goes into paged RAM, a piece of memory which occupies the same location as the Spectrum ROM and can be "toggled" in and out just the same as Interface 1 (and *that* doesn't lose programs). The Disciple has a special command to enable you to get straight into that piece of paged RAM and alter directly the system variables that were fed in. It's the POKE @ command, and is extremely powerful. Those who already have Disciple will possibly not be aware of this command, so I have provided a full list in Figure 1. For those who already know about POKE @, there are some extra commands in this list. So make a photocopy of the table and keep it with your Disciple Manual.

Most of the instructions are self-explanatory; for instance, POKE @ 0,n cuts down the histrionic border-flash when the discs are in use to more bearable proportions. Some are a bit more subtle, though. POKE @ 1 or 2 are for drives 1 and 2: POKE @ with 40 or 80 for number of tracks on single sided discs, add 128 to each POKE for double sided discs. POKE @ 3 sets the step rate in milliseconds; what the table does not mention is that any figure you put in, either in the start-up program or by POKEing @ 3 has 6 automatically added to it. POKE @ 6,1 turns tokens off and is essential for setting up Tasword: the extra CHR$ can then be omitted from printer control commands. POKE @ 12 and 13, (12 for the low order byte and 13 for the high order byte) alter the length of the snapshotting file, normally 49512 bytes. So if you just need to snapshot the screen, then POKEing with 6192 should do the trick.

Finally, if you already have a Disciple and would like to contribute useful routines or comments of general interest, please send them in to ZX Computing, marking your envelopes Disciple.

Figure 1

The Poke @ Command

This command allows a value between 0 and 255 to be stored in the variables area of the disc operating system (GDOS).

POKE @ 0,n (n=0 to 8) (RBCC) default 7
The border colour change is essentially the sector number when reading and writing on the disc. This variable is ANDed with the sector value, then output to the screen border. If 0 is chosen, no border colour will occur. If 7 is chosen, all sectors will change the border colour.

POKE @ 1,n (n=40 or 80 + 128) (TRAKS1) default 208
This variable holds the number of tracks and whether double or single sided for disc drive one.

POKE @ 2,n (n=40 or 80 + 128) (TRAKS2) default 208
This variable holds the number of tracks and whether double or single sided for disc drive two.

POKE @ 3,n (n=0 to 255) (STPRAT) default 6
This variable allows you to set the GDOS for the different step rates of drives (usually 12 to 30 milliseconds). The step rate is not allowed to go lower than six milliseconds.

POKE @ 4,n (n=0 or 1) (ZXPNT) default 0
If the setting is 0, then GDOS uses the centronics printer port. If 1, then the printer channel is unchanged by SDOS.

POKE @ 5,n (n=0 to 255) (WIDTH) default 80
This variable sets the number of characters allowed per line when using LPRINT and LLIST with the centronics printer port.

POKE @ 6,n (n=0 or 1) (PCODE) default 0
If the setting is 0, then the printer driver looks for attribute codes such as TAB and AT within LPRINT and LLIST commands. If the setting is 1, then the printer driver outputs directly to the printer the absolute value of the code 0 to 255. This is essential for sending control codes to the printer. E.g:
POKE @ 6,1: LPRINT CHR$ 27; CHR$ 65: POKE @ 6,0

POKE @ 7,n (n=0 to 255) (LSPCE) default 12
This variable sets the printer line feed in increments of n/72 of an inch during graphics printing.

POKE @ 8,n (n=0 to 255) (LFEED) default 1
This variable sets the number of line feeds automatically executed by the printer driver after a carriage return.

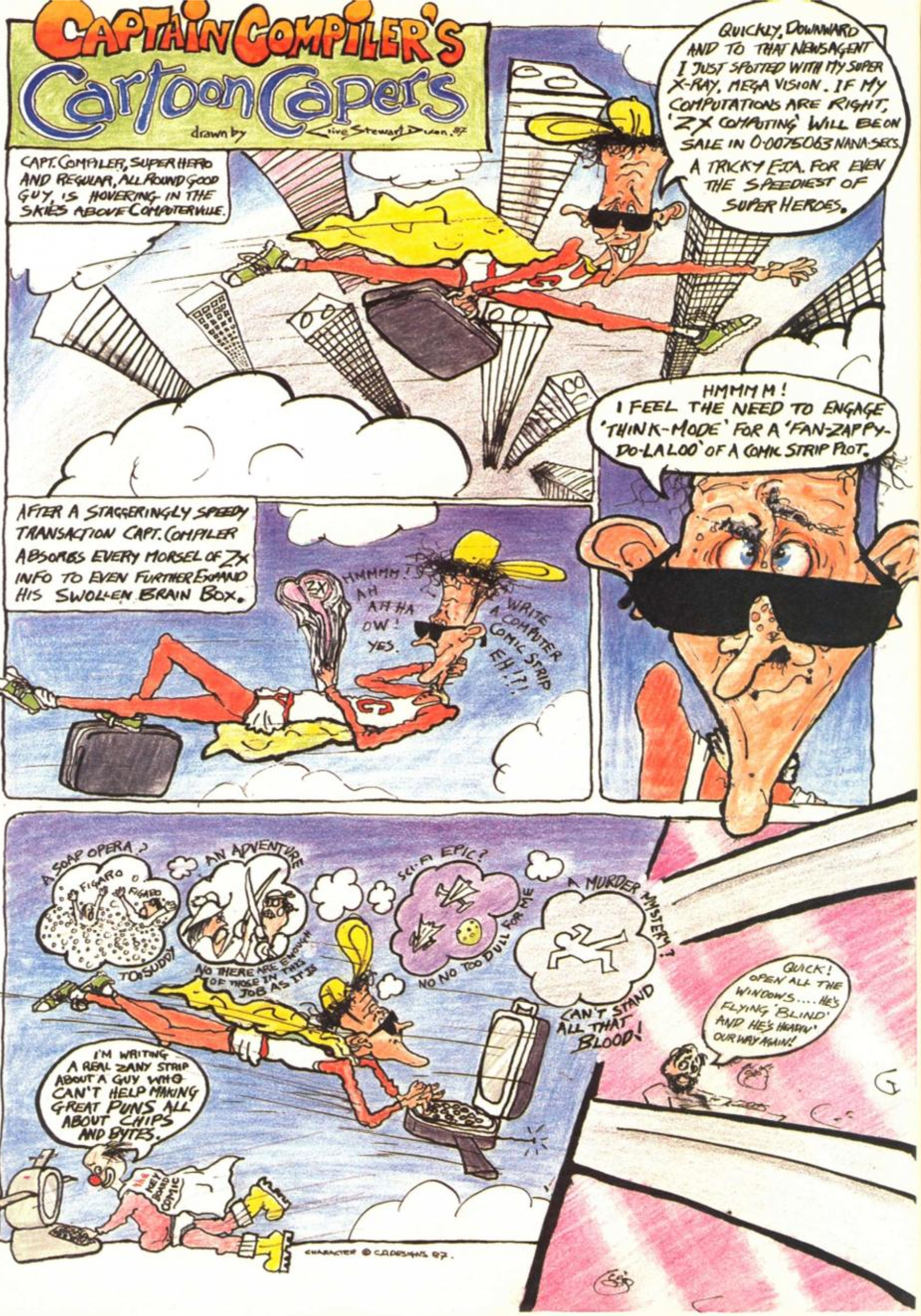
POKE @ 9,n (n=0 to 255) (LMARG) default 0
This variable sets the number of spaces inserted automatically by the printer driver after a carriage return. It is used for left hand margin control.

POKE @ 10,n (n=0 or 1) (GRAPH) default 1
If the setting is 1, the printer driver generates the graphic representation of, and on LPRINT and LLIST statements. If the setting is 0, the normal printer character is output.

POKE @ 11,n (n=1 to 64) (NSTAT) default 1
This variable sets the network station number. Be careful not to poke 0.

POKE @ 12,n (and POKE @ 13,n) (n=0 to 49152) (LSNAP) default 49152
This variable controls the length of file which the snapshot takes: POKE @ 12 with the low order 256 byte and POKE @ 13 with the high order 256 byte.

POKE @ 14,n (and POKE @ 15,n) (n=0 to 65536) (ONERR) default 0
If the setting is 0, then GDOS returns to the Spectrum on errors which do not relate to hook codes or GDOS syntax statements. If an address is poked into these two locations, it makes a call indirectly to this address for further syntax checking. Necessary when adding extra commands to the syntax.


CAPTAIN COMPILER'S
Cartoon Capers
drawn by Clive Stewart Dixon. 87
CAPT. COMPILER, SUPER HERO AND REGULAR, ALL ROUND GOOD GUY, IS HOVERING IN THE SKIES ABOVE COMPUTERVILLE.
QUICKLY, DOWNWARD AND TO THAT NEWSAGENT I JUST SPOTTED WITH MY SUPER X-RAY, MEGA VISION. IF MY COMPUTATIONS ARE RIGHT, 'ZX COMPUTING' WILL BE ON SALE IN 0·0075063 NANA-SECS. A TRICKY E.T.A. FOR EVEN THE SPEEDIEST OF SUPER HEROES.
AFTER A STAGGERINGLY SPEEDY TRANSACTION CAPT. COMPILER ABSORBS EVERY MORSEL OF ZX INFO TO EVEN FURTHER EXPAND HIS SWOLLEN BRAIN BOX.
ZX
HMMMM! AH AH HA OW! YES.
WRITE A COMPUTER COMIC STRIP EH!?!
HMMMM! I FEEL THE NEED TO ENGAGE 'THINK-MODE' FOR A 'FAN-ZAPPY-DO-LALOO' OF A COMIC STRIP PLOT.
A SOAP OPERA?
FIGARO
FIGARO
TOO SUDDY
AN ADVENTURE
NO THERE ARE ENOUGH OF THOSE IN THIS JOB AS IT IS
SCI-FI EPIC?
NO NO TOO DULL FOR ME
A MURDER MYSTERY?
CAN'T STAND ALL THAT BLOOD!
I'M WRITING A REAL ZANY STRIP ABOUT A GUY WHO CAN'T HELP MAKING GREAT PUNS ALL ABOUT CHIPS AND BYTES.
QUICK! OPEN ALL THE WINDOWS.... HE'S FLYING 'BLIND' AND HE'S HEADIN' OUR WAY AGAIN!
KEY BOARD COMIC
CHARACTER © C.D.DESIGNS 87.

Comic Strip

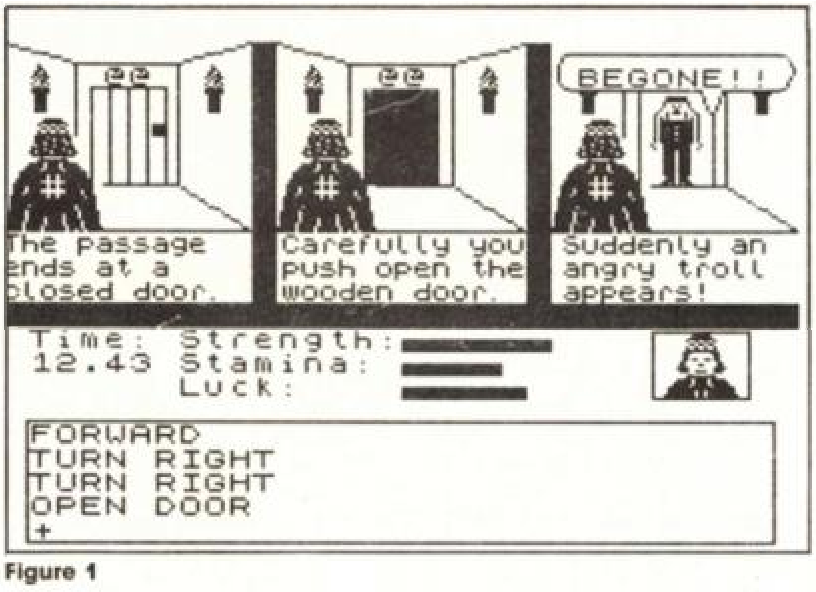


Figure 1

RANDOM MEMORY

Clyde Bish begins a three part series showing how you can produce your own computerised cartoon strip.

The medium of the comic strip has always been a popular one. Hands up anyone who hasn't followed the exploits of Dennis the Menace, Batman, Judge Dredd, or, for more mature ZXC readers, that "strip" heroine, Jane! In fact, PhD's have been awarded on analyses of the exploits of Asterix and Co. It is also a medium which lends itself readily to the computer screen — Melbourne House's Redhawk, for example — so let's switch on the machinery, and the grey matter, and see what facilities are required.

Obviously we need the strip itself, ideally scrolling right to left, with the new illustration being drawn in the space vacated screen right. In addition to the actual picture we will also need captions, and perhaps speech bubbles. We'll also need an on-screen input system to a vertically scrolling communications window, indications of strength, stamina etc. for an adventure (or profits, stocks held etc. for a strategy game) and perhaps a real-time clock and, in a "Superman"-type strip, a visual reminder of your persona at any given time.

With these requirements I came up with a screen display like that shown in Fig. 1. I emphasise that this is just a suggestion — you will be writing the program which uses the strip so you must decide on the final display format, and the purpose of this series is to show you how. (Later in the year I'll be showing you a way to convert adventures from the book/magazine format to the computer so if you're lacking ideas you'll get some help then).

Piccies

Let's start this month with the illustrations. Look at a strip cartoon and you'll see that it's made up of three 'layers'. Furthest back is, not surprisingly, the background. This is usually a fairly simple line drawing with various embellishments such as pictures on a wall or flaming torches as in Fig. 1. In front of this, often in a multi-layer format are the characters. (So we don't confuse these characters with the character symbols we'll be printing as embellishments I'll call the former "icons"). Finally comes the speech bubble.

Bearing this in mind we obviously need a similar system for a computer strip, each element taking up as little memory space as possible. Since a background or icon may be used more than once in a story line it would be uneconomic to store each complete illustration separately. Instead, we'll hold each element as a series of bytes in memory, and call them, more or less as subroutines, when needed by a driver routine.

Let's get the necessary machine code on board first. **CLEAR 64935,** type in the loader (Program 1), setting the *from* and *end* values to **64936** and **65375** respectively, then **RUN** and enter the numbers from Table A, reading across each line. Save the code with:

SAVE "drawCODE" CODE 64936, 440

and **verify.**

Table A

62	2	205	1	22	17	96	234
26	254	0	40	30	79	19	197
26	50	0	91	19	26	79	19
26	71	213	197	209	58	0	91
71	26	215	19	16	251	209	193
13	32	227	19	26	254	0	202
112	254	79	19	197	33	0	64
26	71	213	17	32	0	25	16
253	209	19	26	79	6	0	9
19	26	79	19	26	71	19	26
213	197	209	254	0	40	39	254
1	40	70	6	6	14	6	197
26	182	119	19	35	16	249	213
17	26	0	25	209	193	13	32
238	213	17	64	0	25	209	124
254	72	56	223	24	68	6	3
14	4	197	26	182	119	19	35
16	249	213	17	29	0	25	209
193	13	32	238	213	17	128	0
25	209	124	254	72	56	223	24
33	6	4	14	5	197	26	182
119	19	35	16	249	213	17	28
0	25	209	193	13	32	238	213
17	96	0	25	209	124	254	72
56	223	209	193	13	194	219	253
19	26	254	0	200	17	144	254
237	83	0	91	50	4	91	205
173	254	62	22	215	62	2	215
58	4	91	215	62	144	215	201
184	171	247	171	250	170	252	168
253	165	250	160	247	159	184	159
181	160	179	162	178	165	179	168
181	170	184	171	0	237	91	0
91	26	103	19	26	71	62	175
152	111	19	26	71	19	213	26
87	62	175	154	79	205	214	254
209	19	26	254	0	200	254	1
32	234	19	26	24	220	197	205
48	255	193	17	1	1	120	148
48	4	21	21	237	68	71	121
149	48	4	29	29	237	68	79
176	200	121	184	229	98	107	34
2	91	46	0	56	4	101	107
72	71	34	0	91	96	120	203
63	111	125	129	56	3	184	56
8	144	111	237	91	2	91	24
5	111	237	91	0	91	227	124
130	103	125	131	111	197	205	48
255	193	227	37	32	220	225	201
125	230	192	31	55	31	15	173
230	248	173	87	124	7	7	7
173	230	199	173	7	7	95	124
230	7	71	4	62	254	15	16
253	71	26	160	168	47	18	201
129	66	66	66	36	36	36	24

Comic Strip



Drawing

Let's begin with an explanation of the background drawing routine. This, remember, is made up of two elements, the line drawing, and the embellishments (torches and mystic symbols over the door in my example screen). The line drawing data is held as a series of PLOT or DRAW co-ordinates. Each set starts with a PLOT pair and continues with a number of DRAW pairs. The DRAW co-ordinates are *absolute*, i.e. the actual co-ordinates of the pixels you want to draw to, not offset values as in the Basic DRAW. This makes inputting values from a drawing on pixel paper much easier. When the routine finds a 1 then next pair starts a new PLOT position. A terminal 0 is needed to return from the routine.

To use this element, the data (PLOTs, DRAWs, 1s and 0s) are held in a series of bytes starting at a known address, say 50000, and the routine is called with a line such as:

10 POKE 23296, 80: POKE 23297, 195: RANDOMIZE USR 65197

The POKEs (into the first two bytes of the Printer Buffer) are the low and high values of the data start address (80+195*256=50000). Look at the beginning of Table B, the data we'll use for the demonstration at the end and you'll see the 1's which reset the PLOT position and the 0 to return.

The remainder of the elements are called with a single USR call. Don't try to link these routines with the background in one single call. The routine will crash! This call produces the embellishments, the icons and the speech bubble. The information to control the printing (but not the data itself) is held in a string of numbers as follows:—

a) Embellishments — A 0 to start means miss out this element. Any other number is the total number of this type of element to be printed, so in Fig. 1 this would be '3' as there are two torches and the mystic symbol block to print. There then follows a series of numbers in blocks of three. The first of these is the length of that data block (maximum 255), which begins at the address given by the next two numbers. So the sequence:

3,7,80,195,10,180,195,6,200,195

would print three objects. The first is made up of 7 bytes of data beginning at 80+195*256=50000. The remaining numbers would translate in a similar way.

b) Icons — As before a starting 0 means miss out this element. Any other number is the total number of icons to print. (For Fig. 1 this would be '2' — one for the troll and one for the foreground character). This number is followed by others in blocks of five. The first number in each block is the Print At row, the second the column, the third and fourth the address (held as before) and the fifth is the type of icon shape. There are three types of icon available. Type 0 is three character squares wide by four high, for small characters like the troll. Type 1 is four wide by five high, and type 2 is a 6x6 grid for close-ups (or two small

PROGRAM 1

```
  10 FOR f=from TO end: INPUT i:
POKE f,i: PRINT f,i: NEXT f
  20 REM alter from & end number
        as required
```

1 13 25 37 49 61 73 85	2 14 26	3 15 27 39 51 63 75 87
4 16 28 40 52 64 76 88	5 17	6 18 30 42 54 66 78 90
7 19 31 43 55 67 79 91	8 20	9 21 33 45 57 69 81 93
10 22 34 46 58 70 82 94	11 23 95	12 24 36 48 60 72 84 96

Figure 2

characters which always appear together). Each is printed in such a way as not to obliterate the background where no icon pixels are being filled. The sequence:

2,1,4,80,195,0,2,10,180,195,2

would print two icons, the first a 3x4 block with top left corner at row 1, column 4, from data starting at 50000. I'll leave you to work out the second. Note that this mode only operates within the top third of the screen. If the bottom of a block comes below row 7 strange things will happen!

c) Speech Bubble — Again, 0 means miss out, otherwise the first (and only digit) refers to the column where the pointer *V* is printed. (In Fig. 1 it's at column 28). The start of such a control string is POKEd into 64942 (low byte) and 64943 (high byte) before calling the routine with **RANDOMIZE USR 64936.**

Data

Now to the actual data to which the control string refers.

a) Embellishments — This is held as:

22, row, column, codes of characters to be printed (see Appx. A of your Spectrum manual) resetting with *22, row, column* whenever a new print position is required. So the data string:

22,1,0,48,49,50,22,2,5,65,66

(which, incidentally, has a length of 11 — you count the 22s as well), would print "012" at row 1, column 0 and "AB" at row 2, column 5. You can include block graphics or any user-designed graphics you have made (20 are available as *A* is used for the bubble pointer). You can also include control characters (see p.183 in the manual), so 16, 2 would set INK to red, 17, 6 would print on yellow paper, 19, 1 would have the colours bright, and 18, 1 would have them flashing. (Try 22, row, column, 18, 1, 134, 137 for minor "animation"!)

b) Icon data — The data for these is the actual bit patterns for each screen byte to be used. Each character square on screen is made up of eight lines of eight pixels one below the other, but held in the computer's memory one *after* the other! To make matters worse they are held in a rather strange order. To see what I mean enter the line

FOR f=16384 TO 18432: POKE f, 255: PAUSE 20: NEXT f

and watch the way the screen fills. What this amounts to is, apparently, nothing less than confusion! It isn't really. Refer to Fig. 2 and you'll see that you have to POKE in the bit pattern of each byte (using the BIN function) in the order given for a 3x4 icon. (If you're still confused wait until next month and I'll supply an editor program to do the job for you).

To end, let's pull all the ideas together with a grand demonstration of the facilities offered. We'll reproduce the top right picture from Fig. 1 but to save on typing we'll just use one icon, the troll, and the speech bubble will remain empty (we'll deal with how to fill that later).

Alter and RUN the loader program to enter the data from Table B from 50000 to 50131. Enter just the numbers, not the text lines, those are to help you find your way around. Now alter the loader again to enter the control string data (Table C) into addresses 60000 to 60010.

We're almost ready to roll, but we need a line to call up the routines. Type in Program 2 and RUN for the display. The REMs will explain what each part does.

Now, what about that empty speech bubble. There are eight character squares within it, so provided you only need that number of spaces as a maximum you could just **PRINT AT 1, 23; "text"** (like the "BEGONE!! in Fig. 1). That doesn't help if, as is more usual, you want to display more information. Program 3 will come to your aid. It is a basic routine for scrolling text through a window.

Table B

```
Background data

176 175 199 168 232 168 232 127
199 127 199 168 1   233 168 255
175 1   232 127 255 112 1   199
127 176 112  0

Embellishment data

22  2   23  138 22  2   30  133

Icon data

0   0   0   68  17  0   95  125
0   15  120 0   0   0   0   84
21  0   95  253 0   15  120 0
7   240 0   95  253 0   143 248
128 15  120 0   7   240 0   95
253 0   79  249 0   15  120 0
8   8   0   95  125 0   15  120
0   15  120 0   9   72  0   95
253 0   15  120 0   7   112 0
16  132 0   95  125 0   15  120
0   15  120 0   37  210 0   95
253 0   15  120 0   29  92  0
```

It takes the message string m$, adds eight blanks spaces to the front (to make the bubble start empty) then uses a loop, set to the original length of the string to print successive sections of the new string in the speech bubble. Pressing SPACE at any time during the scroll will restart it. Add to it your driver (Program 2) and try it out.

Well, that's stage one in the can. Watch this space for further developments.

Table C

```
P_torch

1  8  108  195

P_icon

1  2  26  116  195  0

Bubble pointer

28
```

PROGRAM 2

```
   5 REM drawCODE must be on
        board
  10 POKE 23296,80: POKE 23297,1
95: RANDOMIZE USR 65197: POKE 64
942,96: POKE 64943,234: RANDOMIZ
E USR 64936
  20 REM draw background then
        remainder
```

PROGRAM 3

```
  50 LET m$="THIS IS AN EXAMPLE
OF SCROLLING IN A SPEECH BUBBLE"
: LET m=LEN m$: LET m$="        "
+m$
  60 FOR f=1 TO m: PRINT AT 1,23
;m$(f TO f+7): PAUSE 10: IF INKE
Y$=" " THEN  GO TO 60
  70 NEXT f
```

SHORTCUTS

More compact classics from ZXC readers.

Another mixed bag of programs which prove the old saying about good things coming in small packages. This month we start with a program which will hopefully slightly ease the burden of questions to Crosswires.

It is a program which will be greeted warmly by all those who purchased a Dixons 128K/Printer package, there are still a lot of problems to solve but this is one less, and so it gets the coveted STAR CUT award.



48K RS232

Chris Richards of Huddersfield supplies the solution to getting the printer to work in 48K mode and all you have to do is type in his listing and run it.

Provided no error report is produced, SAVE the code using **SAVE "RS232" CODE 65000,113.** To use it make sure you have reloaded it each time and type **RANDOMIZE USR 65000** and all subsequent LPRINTING will be done via the RS232.

One problem is that no line feed is sent at the end of a line, so you will have to set your printer's DIP switches to produce an automatic line feed after a carriage return. If this is not possible then you are limited to hand cranking the machine at the end of each line and LLISTing is likely to become a little messy.

The BAUD rate is automatically set to a default of 9600. If you require another setting then one way is to FORMAT as usual in 128 mode and make a note of the values at addresses 23391/2 by PEEKing them, and then in 48 mode POKE them into addresses 23728/9.

```
   1 REM 48K RS232
   2
  10 CLEAR 64999
  20 LET c=0
  30 FOR f=65000 TO 65112
  40 READ a: LET c=c+a: POKE F,A
  50 NEXT f
  60 IF c<>14262 THEN  PRINT AT
0,0;"ERROR in DATA": BEEP 3,-12
  70 DATA 42,79,92,237,91,28,92,
25,43,17,254,253,115,35,114,33,1
1,0,34,176,92,201: REM End of pa
rt 1!
  80 DATA 245,14,253,22,255,30,1
91,66,62,14,237,121,205,75,254,2
37,120,230,64,32,247,42,176,92,1
7,2,0,183,237.82,235,241,47.55,6
,11,243,197
  90 DATA 245,62,254,98,107,1,25
3,191,210,53,254,230,247,237,121
,24,6,246,8,237,121,24,0,43,124,
181,32,251,0,0,0,241,193,183,31.
16,218,251,201
 100 DATA 62,127,219,254,31,216,
62,254,219,254,31,216,207,12
```

Stripes

Barry Stuart, a resident of Liverpool, sent us one of those programs which falls into the "not-too-sure-what-I-can-use-it-for-but-it-might-come-in-handy-one-day" category.

The program draws several broad stripes across the screen then revolves them vertically at a speed selected by yourself, very pretty and possibly of use to hypnotists.

```
   1 REM  stripes
   2
  10 LET c=0: FOR i=32768 TO 328
68: READ a: POKE i,a: LET c=c+a:
 NEXT i
  20 IF c<>11633 THEN  PRINT "da
ta error!": STOP
  30 INPUT "Speed 1 TO 256 ";s:
POKE 32773,s-1
  40 PAUSE 1: PAUSE 10: RANDOMIZ
E USR 32768
  50 DATA 175,111,24,52,38,49,12
5,60,230,7,111,125,95,22,8,118
  60 DATA 211,254,1,2,226,16,254
,13,32,251,21,40,23,123,60,230
  70 DATA 7,95,1,3,30,16,254,13,
32,251,211,254,219,254,47,230
  80 DATA 31,192,24,230,37,32,21
2,125,198,7,230,7,229,243,237,11
5
  90 DATA 176,92,49,0,91,22,6,79
,135,135,135,129,103,111,6,16
 100 DATA 229,229,229,229.16,250
,121,61,230,7,21,32,234,237,123,
176
 110 DATA 92,225,251,24,159
```

Shadow Effect

Screen presentation is a common preoccupation and this routine produces an unusual and pleasing effect. Gary Franklin of Canvey Island wrote it and once the machine code listing has been entered, run and saved, then it will shadow anything on the screen printed with BRIGHT 1.

Experiment with the lines as suggested to produce your own individual effect.

```
   1 REM SHADOW EFFECT
   2
   5 CLEAR 63999: RESTORE : LET
tot=0
  10 FOR x=64000 TO 64210: READ
a: POKE x,a: LET tot=tot+PEEK x:
 NEXT x
  20 IF tot<>27184 THEN  PRINT t
ot;" is wrong! Check data.": BEE
P 1,0: STOP
  30 SAVE "SHADOW"CODE 64000,210
  50 DATA 243,253,229,253,33,255
,90,6,24,197,6,32,197,253,126,0,
254,64,56,15,253,126,1,254
  60 DATA 64,220,82,250,253,126,
32,254,64,56,93,253,43,193,16,22
8,193,16,222,253,225,251,201,122
  70 DATA 230,3,7,7,7,246,64,103
,107,201,36,124,230,7,192,124,21
4,8,103,125,198,32,111,208
  80 DATA 124,198,8,103,238,88,1
92,38,0,201,120,254,32,200,253,2
29,209,19,205,47,250,221,33,191
  90 DATA 250,6,8,62,64,253,190,
33,56,9,6,10,253,190,32,48,2,6,1
2,221,126,0,182,119
 100 DATA 221,35,205,58,250,16,2
44,201,225,241,245,229,254,24,21
0,35,250,253,229,209,235,1,32,0
 110 DATA 9,235,14,2,221,33,203,
250,205,47,250,6,4,221,126,0,182
,119,221,35,205,58,250,16
 120 DATA 244,253,126,33,254,64,
210,35,250,241,245,254,32,210,35
,250,19,13,32,220,195,35,250
 124
 125 REM Change shadow data
below if desired.Vertical shade
8,10 or 12 pixels depth * 8
pixels width.
 126
 130 DATA 0,0,128,64,160,80,160,
80,160,80,160,80
 134
 135 REM Horizontal shading 4
 pixels depth * 8 or 16 pixels
depth.
 136
 140 DATA 42,21,10,5,128,64,160,
80
 200 PRINT "Press key for demo":
 PAUSE 0
 210 BORDER 6: PAPER 6: INK 0: C
LS : PRINT AT 10,5; BRIGHT 1; IN
VERSE 1;" ZX COMPUTING MONTHLY "
 220 FOR X=0 TO 20 STEP 4: PRINT
 AT X,4; BRIGHT 1; PAPER RND*7;
INK 9;" FOR THE BEST PROGRAMS! "
 250 RESTORE USR 64000
 300 NEXT X: PAUSE 0
```

Biggies

This is a bit of a mystery which I thought I'd include as a challenge to computer sophisticates.

The program enlarges and prints characters to the screen in a 2x2 size. The documentation was non-existent but the program produced a pleasant and I can tell you that changing the DATA of line 110 changes the message. A=33, X=58.

Handy for fast, short headings but if you want to extend the message length or change the printed screen position then you'll have to get your disassemblers out!

```
   1 REM biggies
   2
  10 LET s=0: FOR f=50000 TO 500
93: READ a: POKE f,a: LET s=s+a:
 NEXT f
  20 IF s<>9471 THEN  PRINT "Dat
a error": STOP
  30 RANDOMIZE USR 50000
 100 DATA 1,162,195,33,36,64,17,
0,61,229,10,38,0,111,41,41,41,25
,235,225,197,229,6,2,197,6,4,197
,205,138,195,36,205,138,195,36,1
9,193,16,243,1,224,7,237,66,193,
16,232,225,35,35,193,3,203,117,1
92,24,204,229,26,6,4,23,203,22,2
03,22,203,78,40,2,203,198,16,243
,35,203,69,32,236,225,201
 110 DATA 58,56,0,35,47,45,48,53
,52,41,46,39
```

Mini-label

A. Welsh from Glasgow sent this small program to produce a small label strip on the printer which you then cut out and stick to the cassette or place between the edge of the cassette insert and the plastic case.

The S — — — F prompt provides an easy way of formatting the layout of the output line.

```
   1 REM mini label
   2
  10 BORDER 1: INK 7: PAPER 1: C
LS : LET c$="--------------------
---------------"
  20 INPUT "S    -    -    -    -
  -    -    F"; LINE a$
  30 INPUT "S    -    -    -    -
  -    -    F"; LINE b$
  40 PRINT c$;a$: PRINT : PRINT
b$: PRINT c$
  50 PRINT AT 12,0;"PRESS P TO M
AKE A PRINTER COPY  PRESS R TO T
RY AGAIN"
  60 PAUSE 0
  70 IF INKEY$="P" OR INKEY$="p"
 THEN  GO TO 90
  80 IF INKEY$<>"P" OR INKEY$<>"
p" THEN  GO TO 10
  90 LPRINT c$;a$: LPRINT : LPRI
NT b$: LPRINT c$: CLS : GO TO 20
testing zxc
```

Attributes

Another 128K specific program, this one from Settle programmer D.J. Walker changes the colours of the 128's edit screens. A useful program for those who find the defaults a bit eyestraining.

The code can be located anywhere except in the top page and the destination address for your code is required to be input at the start of the program.

```
   1 REM 128 attributes
   2
  10 INPUT "Load Address? <49128
 ";addr
  40 INPUT "Main Paper ";mp,"Ink
 ";mi
  50 INPUT "Aux Paper ";ap,"Ink
";ai
  60 LET aux=ai+8*ap
  70 LET main=mi+8*mp
 110 RESTORE : FOR n=0 TO 23
 120 READ a: POKE (addr+n),a
 150 NEXT n
 170 DATA 243,62,23,1,253,127,23
7,121,33,15,236,54,aux,33,17,236
,54,main,62,16,237,121,251,201
 180 RANDOMIZE USR addr
```

Feedback

A nice feature of this section is the lively correspondence we get in response to the programs we print. HEX/DEC in our February issue drew a reply from W.E. Thomson of Aldeburgh, who comments on the ingenuity of the previous program then offers his version at a third of the length!

DEC-HEX makes good use of the FN function, one calling the other and also employs the technique of splitting a program into simple steps.

Alan Knight of Bromley writes that Ben Stragnell's December routine for producing 64 columns to a line won't work in 128 mode, this is because UDG "u" is used and the 128 also needs it as a play token. He continues that the assumption that the UDG's are at the top of memory suffers when you use other code in this area and have perhaps relocated the UDGs. (Or even if you have a 16K machine).

Finally, should you attempt to print a character with a code of less than 32 (by mistake) then the program crashes. The test Ben made tended to disrupt the stack. However, not being one to merely offer negative criticism, Alan has provided a rewritten version which allows for all these things. We print it as HEX dump and you can replace the code in the December issue with it or enter it with a loader program to address 30000. It is 154 bytes long.

If you replace the old printed code then you will have to change line 130 to:

130 SAVE N$ CODE 30000,154.

```
   1 REM DEC-HEX
   2
  20 DEF FN h$(n)="0123456789ABC
DEF"(n+1)
  30 DEF FN b$(n)=FN h$(INT (n/1
6))+FN h$(n-16*INT (n/16))
  40 INPUT "Decimal value (0-655
35):";d
  50 IF d<>INT d OR d<0 OR d>655
35 THEN  GO TO 40
  60 LET hi=INT (d/256): LET lo=
d-256*hi
  70 PRINT d;"(dec)=";lo;",";hi;
"(2-byte dec)="'FN b$(hi);FN b$(
lo);"(hex)": GO TO 40
```

64 Revisited

```
2A 7B 5C 11 90 00 19 E5
DD E1 2A 4B 5C 7E FE D3
C0 01 06 00 09 56 23 5E
23 7A FE 40 38 03 16 00
1C 7B FE 16 D0 7E FE 80
C8 FE 20 D8 E5 D5 18 05
D1 E1 14 18 E3 D6 20 26
00 6F 06 03 CB 25 CB 14
10 FA 11 00 3D 19 DD E5
D1 01 08 00 ED B0 DD E5
E1 06 08 7E 4F 17 B1 4F
3E 00 C5 06 04 CB 19 CB
19 1F 10 F9 77 C1 23 10
EA D1 D5 7A CB 3A 30 13
DD E5 E1 BF 06 08 7E C5
06 04 CB 3F 10 FC 77 C1
23 10 F3 3E 15 D7 3E 01
D7 3E 16 D7 7B D7 7A D7
3E A2 D7 3E 15 D7 3E 00
D7 18 95
```

EXPERT SYSTEMS

Part 3: David Nowotnik shows how you can turn your computer into an 'expert'.

It's reassuring to know that most of us, in our own right, are experts. If you are working, and if you've been doing that job for some time, then you are an expert. Anyone starting to do the same job as you should regard you as an expert, and they should, if they have any sense, be tapping your experience to enable them to become experts too. And for those who are too young to be experts, then you are probably training to become skilled in one or more areas; you are the experts of the future.

If you've been following this series, then you are learning to become an 'expert' in expert systems. At present, I hope that you regard me as an expert in this area, because I am able to pass on to you experience and knowledge to allow you to gain some expertise in this area. But, expert systems are a long way from my principle area of expertise, and there are now many expert system professionals from whom I could learn a great deal.

All of which goes to show that expertise is relative; someone who knows a little about a subject will appear an expert to someone who knows nothing at all about that area. Experts should always be learning more about their subject, to become better experts. This is even true for 'high-powered' experts; rules and procedures will change, knowledge in any field is constantly expanding, and all experts, to maintain their expertise, must learn these new rules and knowledge.

The same is also true of computer expert systems. With the help of a human expert, they must be capable of learning new rules and knowledge to become better experts, and adjust to changing circumstances. How expert systems can learn is the subject of this, the third part of my expert system series.

Facts!

One of the principle ways in which we learn is simply to take in facts. Experts give us facts through books, TV, newspapers, (and ZX Computing, of course!) etc, as well as face-to-face as teachers, lecturers and tutors. Given facts in this way, our brain then files them away for future use.

In the expert systems I gave you last month, I gave those programs facts, as well as the means of storing and recalling those facts. I 'taught' the computer those facts by including that information in DATA lines in the BASIC programs. These expert systems could be taught more facts through a programmer adding more DATA lines containing





Fig 1. Learning Example in Serial Decision making

```
100 REMark Simple Expert System
110 REMark  Learning new facts
120 REMark  (Serial Decisions)
130 REMark
140 REMark David Nowotnik
150 REMark February, 1987
160 REMark
170 REMark Initialise
180 DIM q$(50,40): DIM l(50)
190 DIM r(50,2): LET n=0: RESTORE
200 FOR i=1 TO 7
210 READ p$: LET l(i)=LEN (p$): LET q$(i)=p$
220 FOR j=1 TO 2
230 READ r(i,j)
240 NEXT j
250 LET n=n+1
260 NEXT i
270 DATA "Is it a home computer?",2,7
280 DATA "Does it have at least 128K of RAM?",3,6
290 DATA "Has it a built-in tape recorder?",5,4
300 DATA "Spectrum 128",0,0
310 DATA "Spectrum +2",0,0
320 DATA "Spectrum Plus",0,0
330 DATA "IBM PC",0,0
340 :
400 CLS
410 PRINT " Expert System for Computer": PRINT " Selection"
420 PRINT: PRINT: PRINT " Select:": PRINT
430 PRINT " 1. Use the system"
440 PRINT " 2. Expand knowledge base"
450 PRINT " 3. Stop"
460 PRINT "      ";: INPUT d$
470 IF d$="1" THEN GO SUB 500: PRINT: GO TO 410
480 IF d$="2" THEN GO SUB 800: PRINT: GO TO 410
485 IF d$="3" THEN GO TO 2000
490 GO TO 450
500 REMark Use the expert system
510 LET x=1: PRINT
520 LET a$=q$(x): LET a$=a$(1 TO l(x))
530 PRINT a$;" ";: INPUT k$
540 IF k$="y" OR k$="Y" THEN LET x=r(x,1): GO TO 560
550 LET x=r(x,2)
560 PRINT
570 IF r(x,1)=0 THEN GO TO 600
580 GO TO 520
590 :
600 REMark Give the answer
610 PRINT "The answer is a ";q$(x)
620 RETurn
630 :
800 REMark Teach the system new facts
810 PRINT: PRINT: IF n>48 THEN PRINT "Knowledge base full":
RETurn
820 PRINT "First use the system until we get "
830 PRINT "to the end of a branch that you wish"
840 PRINT "to alter."
850 GO SUB 500
860 PRINT : PRINT "Do you want to add to this point? (y/n) ";
870 INPUT k$
880 IF NOT (k$="Y"OR k$="y") THEN RETurn
890 LET n=n+1
900 LET q$(n)=q$(x): LET l(n)=l(x)
910 LET r(n,1)=0: LET r(n,2)=0
920 PRINT : PRINT
930 INPUT "Enter new question: ";t$
940 IF LEN (t$)>40 THEN PRINT : PRINT " Too long ": PRINT: GO
TO 930
950 IF t$="" THEN PRINT : GO TO 930
960 LET q$(x)=t$: LET l(x)=LEN (t$)
970 PRINT
980 INPUT "Enter new answer: ";t$
990 IF LEN (t$)>40 THEN PRINT: PRINT " Too long!": GO TO 970
1000 IF t$="" THEN GO TO 970
1010 PRINT
1020 PRINT "Is ";t$;" the yes or no answer to:"
1030 LET a$=q$(x): LET a$=a$(1 TO l(x)): PRINT a$
1040 PRINT
1050 INPUT "Y or N ";k$
1060 IF k$="y" OR k$="Y" THEN GO TO 1100
1070 IF k$="n" OR k$="N" THEN GO TO 1110
1080 GO TO 1050
1090 :
1100 LET r(x,1)=n+1: LET r(x,2)=n: GO TO 1120
1110 LET r(x,1)=n: LET r(x,2)=n+1
1120 LET r(n,1)=0: LET r(n,2)=0
1130 LET n=n+1
1140 LET q$(n)=t$: LET l(n)=LEN (t$)
1150 LET r(n,1)=0: LET r(n,2)=0
1160 RETurn
1170 :
2000 REMark Save Knowledge base
2010 PRINT
2020 PRINT "Do you want to save the knowledge base?"
2030 INPUT "(y or n) ";k$
2040 IF k$="n" OR k$="N" THEN STOP
2050 IF NOT (k$="y" OR k$="Y") THEN GO TO 2030
3000 REMark QL SAVE DATA routine
3010 DELETE mdv2_expert_base
3020 OPEN_NEW #4, mdv2_expert_base
3030 PRINT #4,n
3040 FOR i=1 TO n
3050   PRINT #4,q$(i): PRINT #4,l(i)
3060   PRINT #4,r(i,1): PRINT #4,r(i,2)
3070 END FOR i
3090 CLOSE #4
3100 STOP
3110 :
4000 REMark Spectrum SAVE routine
4010 SAVE "Expert1": GO TO 400
4020 :
4030 REMark QL LOAD procedure
4040 DEFine PROCedure LD
4050 DIM q$(50,40),l(50),r(50,2)
4060 OPEN_IN #4, mdv2_expert_base
4070 INPUT #4,n
4080 FOR i=1 TO n
4090   INPUT #4,a$,b,c,d
4100   q$(i)=a$: l(i)=b
4110   r(i,1)=c: r(i,2)=d
4120 END FOR i
4130 CLOSE #4
4140 GO TO 400
4150 END DEFine LD
```

information and rules. This is not particularly convenient; it assumes that the expert teaching the system new knowledge is also a programmer. Not only has he to type in DATA lines containing facts, he has to understand how the program works, and the structure of the rules. He may even have to modify or update some of the existing rules or some program lines to allow the system to work with the new rules and knowledge.

So, why have a computer if it can't do things automatically for you. And that's what the first two programs this month will do. One for serial decisions (fig. 1) and one for parallel decisions (fig. 3). Like last month, they were written on the QL, but tested on both Spectrum and QL. They are a little light on the interrogation side; the theory and practise of that was covered last month. These programs aim to show you how to teach facts and rules to an expert system.

Let's start with the serial decision maker in fig. 1. Type in the listing as shown if you have a QL. *If you are using a Spectrum, then omit lines 3000 to 3110 and 4030 to the end.*

All the expert systems this month deal with micro-computers. The idea is that you specify a set of features, and the expert system will select a computer for you which meets your specifications.

For the serial decision example, a small knowledge base is included, which allows you to select one of the Spectrum computers. The decision tree for this very small expert system is shown in fig. 2(a). When you RUN this program, the knowledge and rule arrays are created (line 180) and filled (lines 190 to 260). The questions and answer strings are entered into array q$ (line 210). Numeric array 'r' (line 230) is the rule array which contains the pointers to positions in array q$ depending on 'yes' or 'no' answers. A zero in the rule array indicates that an answer has been reached.

Rules

The rule and knowledge arrays are dimensioned to 50 rules and question/answers, so allow for a reasonable expansion to the seven included (lines 270 to 330 and fig. 2(a)) in the program. On RUNning, once the arrays are set up, the program's basic menu is shown; a choice of interrogating the knowledge base (1), expanding the knowledge base (2) or stopping (3) is given. Using the knowledge base follows much the same idea as last month's program. You give yes/no answers to each question until an answer is given, then you go back to the menu. Before you try teaching the knowledge base anything new, try using the system a few times.

Once you have the feel of the system, press '2' on the main menu. Apart from the introductory message (lines 820 to 840), the first part of adding to the knowledge base is just like using it. The purpose is to get to the appropriate part of the knowledge base that you wish to expand.

If the diagram in fig. 2(a) looks like an upside-down tree to you, then that's exactly what it is — a decision tree. And something like a real tree, this

one 'grows' by spreading new branches from its tips. Once you reach a tip (ie. an answer), you can replace that answer with a question. One answer to that question should be the answer you have just displaced; the other answer, for this expert system, another name of a computer. So the question should be carefully chosen (by you, the expert) to discriminate between the old answer (computer system) and the new one.

Fig. 2b gives you one example. Start from the main menu, press option '2', then select 'y','y','n' to the three questions. You'll end up being told that the answer is a Spectrum 128, and the questions "Do you want to add to this point?" A 'n' reply would take you back to the main menu (an escape route if you've taken the wrong branch); you should reply 'y'. So far we have a home computer, with 128K of RAM, which does not have a built-in cassette deck. This fits a Spectrum 128; it's also true for a QL, and that's the extra answer that I want to insert at this point. So, what yes/no question will differentiate between a QL and Spectrum 128? I've chosen 'Does it have built-in microdrives?'.

So, when the message 'Enter new question' appears on the screen, type in the above question (with the question mark), then enter the new answer, 'QL', on request, and finally, tell the computer whether the answer given is the yes or no response to the new question. The program then sorts out the rule and knowledge base (lines 1100 to 1160) to insert the new question and two answers in the right place. When all that is done in an instant, the program returns you to the main menu.

Knowledge base

In the same way, you can add further knowledge and rules to the system, up to a maximum of 50. If you had answered 'n' to the first question "Is it a home computer?", you would get the response "IBM PC". Now, while that isn't too far from the truth, I'm sure that those computer experts amongst you would want to add a great deal more!

Once you've expanded the knowledge base, the last thing you'll want to do is lose all that work when you turn off your computer. So, when you select 'stop' from the main menu, you get the option to save the knowledge base. If you say yes, then, for the first time, there is a difference between QL and Spectrum versions.

On the Spectrum, (lines 4000 to 4020), the program is simply SAVEd (with all its data), and set to autorun the next time you load, without re-dimensioning the variables.

You can't do that on the QL, so the routine (lines 3000 to 3100) is used instead. Place a cartridge in mdv2_ before saying 'y' to 'Do you want to save the knowledge base?'. A file called *expert_base* is saved on mdv2_ with all the data. Next time you want to use this knowledge base on the QL, load the program, and type in LD instead of RUN. This activates the routine (lines 4030 to 4150) to load in the data from expert_base, before you use the program.

It may have occured to you in using this program that a serial approach is probably not the best way to select a computer. You are more likely to have a list of features that you want, and match those against various computer profiles for the best fit — in other words, use a parallel approach to decide on your

EXPERT SYSTEMS

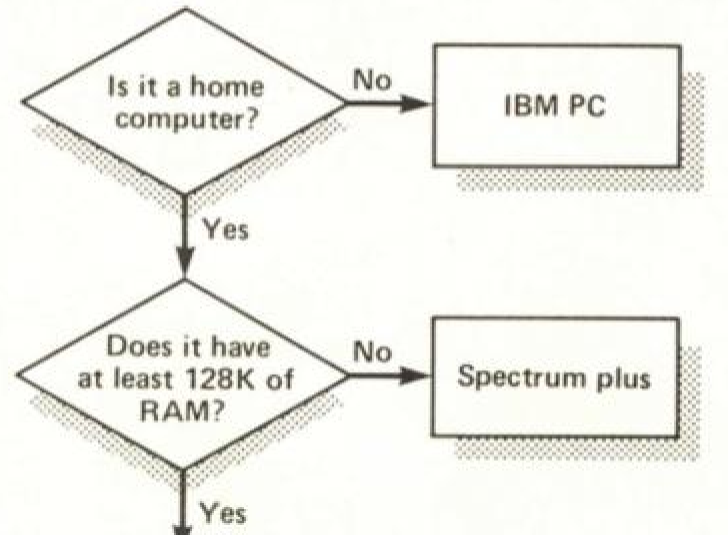


Fig. 2a. Decision tree for serial learning example



Fig. 2b. Adding a new branch to the tree

preferred computer. So, the next teaching program does just that.

Fig.3. contains the listing for this program. The program can be entered as listed on the QL. On the Spectrum, do not type in lines 1060 to 1170 and 1220 to 1370. Also on the Spectrum, the string joining '&' in lines 2060 and 2070 should be '+'.

Just to remind you, in parallel decisions you are asked a series of questions, and your yes/no replies are coded '1' and '0', respectively. This string of zeros and ones is then matched against the zeros and ones in the rule base for all possible answers. If a perfect match is found, then the computer should report this. If the match isn't perfect, the computer should search for the closest match.

In RUNning this program, you'll see it has several features the same as the serial example. The menu is exactly the same. By selecting '1', you get to try out the system, which starts with only four Sinclair computers in its knowledge base though there is room for fifty.

This expert system will ask fifteen questions which I have selected to differentiate between various computers (lines 210 to 360). I'm sure you computer experts can think of many more, but mine should serve as a suitable example. So, you will always need 15 'y' or 'n' replies to elicit a response.

In teaching the expert system some new computers (option '2'), you are first asked to give the name of the new computer. A check is made to see that the name doesn't already exist (lines 800 to 840), then it will ask you to enter your yes/no replies to all fifteen questions (lines 850 to 860), checking to see that exactly the same profile doesn't already exist in the rule base array (r$). If this is satisfied, then your expert rules are stored in the rule array and the name array (a$).

Like the serial example, the knowledge base can be stored when the stop option (3) is selected. Storing (and re-loading) work in exactly the same way as in the serial learning program example.

D.I.Y.

So far, you've had to help your expert system every step of the way. In the final example this month, we'll try and get the

Fig 3. Example Learning Program — Parallel Decisions

```
100 REMark Simple Expert System
110 REMark  Learning new facts
120 REMark (Parallel Decisions)
130 REMark
140 REMark David Nowotnik
150 REMark February, 1987
160 REMark
170 REMark Initialise
180 DIM f$(15,32): DIM a$(50,14): DIM r$(50,15)
190 LET n=0: RESTORE
200 REMark Features
210 DATA "128K of RAM or more"
220 DATA "IBM compatible"
230 DATA "Large selection of games"
240 DATA "Z80 processor"
250 DATA "Eight-bit processor"
260 DATA "Built-in cassette"
270 DATA "Built-in RS232 port"
290 DATA "Built-in Centronics port"
300 DATA "Built-in joystick port"
310 DATA "RGB output"
320 DATA "Supplied with a monitor"
330 DATA "Supplied with disc drive(s)"
340 DATA "Supplied with games"
350 DATA "Supplied with business programs"
360 DATA "Proper keyboard"
370 REMark Answers
380 DATA "Spectrum 48K","001110000000100"
390 DATA "Spectrum Plus","001110000000101"
400 DATA "Spectrum 128","101110101100101"
410 DATA "Spectrum +2","101111101000101"
420 :
430 FOR i=1 TO 15
440 READ b$: f$(i)=b$
450 NEXT i
460 FOR i=1 TO 4
470 READ b$,c$: a$(i)=b$:r$(i)=c$: LET n=n+1
480 NEXT i
490 :
500 CLS
510 PRINT " Expert System for Computer Selection"
520 PRINT: PRINT: PRINT " Select:": PRINT
530 PRINT " 1. Use the system"
540 PRINT " 2. Expand knowledge base"
550 PRINT " 3. Stop"
560 PRINT "      ";: INPUT d$
570 IF d$="1" THEN GO TO 610: PRINT: GO TO 510
580 IF d$="2" THEN GO TO 750: PRINT: GO TO 510
590 IF d$="3" THEN GO TO 1000
600 GO TO 550
610 REMark Use the expert system
620 GO SUB 2000: PRINT: GO SUB 2120
630 IF p>0 THEN PRINT "Perfect Match found, the answer is:":
PRINT a$(p): PRINT : GO TO 510
640 LET t=0: LET high=0
650 FOR i=1 TO n
660 LET max=0
670 FOR j=1 TO 15
680 IF t$(j)="1" AND r$(i,j)="1" THEN LET max=max+1
690 NEXT j
700 IF max>high THEN LET high=max: t=i
710 NEXT i
720 IF t=0 THEN PRINT "No match found": PRINT: GO TO 510
730 PRINT "Best match is ";a$(t): PRINT: GO TO 510
740 :
750 REMark Teach the system new facts
760 PRINT: PRINT: IF n>49 THEN PRINT "Knowledge base full":
PRINT: GO TO 510
770  INPUT "Enter computer name ";n$
780 IF n$="" THEN GO TO 770
790 LET p=0: LET l= LEN(n$): IF l>14 THEN PRINT "Name too
long!": GO TO 770
800 FOR i=1 TO n
810 LET k$=a$(i): LET k$=k$(1 TO l)
820 IF k$=n$ THEN LET p=i
830 NEXT i
840 PRINT: IF p>0 THEN PRINT "That name already exists":
PRINT: GO TO 510
850 PRINT "For ";n$: GO SUB 2000
860 GO SUB 2120
870 PRINT : IF p>0 THEN PRINT "That profile already exists
as:": PRINT a$(p): PRINT: GO TO 510
880 LET n=n+1
890 LET r$(n)=t$: LET a$(n)=n$
900 PRINT "New item stored": PRINT
910 GO TO 510
920 :
1000 REMark Save Knowledge Base
1010 PRINT
1020 PRINT "Do you want to save the knowledge base?"
1030 INPUT "(y or n) ";k$
1040 IF k$="n" OR k$="N" THEN STOP
1050 IF NOT (k$="y" OR k$="Y") THEN GO TO 1030
1060 REMark QL SAVE DATA routine
1070 DELETE mdv2_expert_base
1080 OPEN_NEW #4, mdv2_expert_base
1090 PRINT #4,n
1100 FOR i=1 TO 15
1110   PRINT #4,f$(i)
1120 END FOR i
1130 FOR i=1 TO n
1140   PRINT #4,a$(i): PRINT #4,r$(i)
1150 END FOR i
1160 CLOSE #4
1170 STOP
1180 :
1190 REMark Spectrum SAVE routine
1200 SAVE "Expert1": GO TO 500
1210 :
1220 REMark QL LOAD procedure
1230 DEFine PROCedure LD
1240 DIM f$(15,32),a$(50,14),r$(50,15)
1250 OPEN_IN #4, mdv2_expert_base
1260 INPUT #4,n
1270 FOR i=1 TO 15
1280   INPUT #4,d$
1290   f$(i)=d$
1300 END FOR i
1310 FOR i=1 TO n
1320   INPUT #4,c$,d$
1330   a$(i)=c$: r$(i)=d$
1340 END FOR i
1350 CLOSE #4
1360 GO TO 500
1370 END DEFine LD
2000 REMark Interrogate
2010 LET t$=""
2020 PRINT : PRINT "Reply 'y' or 'n' to each question ":
PRINT
2030 FOR i=1 TO 15
2040 PRINT f$(i);" ";
2050 INPUT k$
2060 IF k$="Y" OR k$="y" THEN LET t$=t$&"1": GO TO 2090
2070 IF k$="n" OR k$="N" THEN LET t$=t$&"0": GO TO 2090
2080 GO TO 2050
2090 NEXT i
2100 RETurn
2110 :
2120 REMark Find a match
2130 LET p=0
2140 FOR i=1 TO n
2150 IF t$=r$(i) THEN LET p=i
2160 NEXT i
2170 RETurn
```

expert system to do a bit more work for itself. The program for this example is in fig.4. It again uses the parallel decision approach, but this time, it tries to develop the rule base for itself. You'll find it's very inefficient, but the program coding will give you an idea how the expert system will automatically adjust its rules — well, almost automatically; it still needs a little help from us!

To keep things fairly simple, there are just five answers in this self-learning program, with seven features. Line 180 dimensions several arrays. As before, f$ and a$ contain the names of the features and answers, respectively. The rules are now in numeric arrays; array 'r' contains the rules stored in the program (the reason for this will become apparent in a moment), while the array 's' is the rule base which the computer will create by self-learning. Numeric array 'y' is used in decision making; it keeps the score of each possible answer when comparing entered features against the rule base.

When you RUN the program, you again get a Menu with three options. Option 1 just tests the self-learning rule base for you. It will go through each answer in turn, and enter the stored features from the correct rule array (r), then gets the computer to make a decision based on those features using the rule array it is developing (s). When you first RUN the program, and select the test option, you'll note that the rule array will always select 'Spectrum 48K' — it still has a lot to learn!

You can start teaching the system to allow it to adjust the rule array either manually or automatically. Option '2' is the manual approach. You will be asked five times (before returning to the main menu) to think of one of the five answers, and provide the correct yes/no response to each of the seven features. The computer will make a guess; if the guess is right, then the rules are not changed, but if the guess is wrong, then the rules are altered, and the updated rules are displayed.

At this point, some explanation is required as to what is happening during adjustment of the self-learning rule array (s). That array is initially filled with zeros (line 410), that's why the answer was always the same on the very first test. When you enter the profile of any one computer system (lines 2000 to 2090) a string of ones and zeros is set up, representing your sequence of yes/no answers. In lines 2200 to 2300 every element in that string is compared with the corresponding rule (in array 's') for each possible answer. If a match is found, then the score (q) is incremented (line 2260), and the total for each possible answer stored in numeric array 'y'. The highest score indicates the best fit of the entered profile and the rule base, and this is how the answer is selected (line 2410).

If the computer's answer is wrong, then you tell it the correct answer (lines 2420 to 2560), and the self-learning rule array is adjusted. Each rule element in the correct answer is incremented if a 'yes' response was given to a particular feature (lines 2770 to 2790). For the answer given incorrecly (and other possible answers which scored the same as the incorrect answer), the rule element is decremented by one for each feature given 'yes' in the profile supplied by you (lines 2710 to 2760).

All that may seem rather complicated, but if you work through it a few times, I hope you will start to see the logic of what is going on. When the rule array is printed out on the screen, you'll get strings of numbers; these are not just restricted to zeros and ones; you'll get higher numbers, and negative numbers too.

Typing in features time and time again is rather boring, so there is an automatic feature entry to speed things up; option 3 from the main menu. It selects a possible answer at random (lines 2100 to 2170 — please note the different structures in line 2110 depending on which machine, Spectrum or QL, you are using), and feeds the yes/no answers automatically into string t$. The computer will then select an answer, as before, using its developing rule array (s). As the computer now knows the correct answer (as it selected this itself!), it tells you if the answer is right or wrong, and updates the rule array if it is wrong.

Whether you use the manual or automatic profile entry procedure, you may be surprised by the number of times this expert system has to 'practise' to pass the test (option 1) and get the answer right every time for every item. In teaching an expert system, it is always much easier to be able to give the computer the correct rules to enter into its rule base. But self modification of the rules may well be necessary sometimes; for example, a manufacturer may change some of the features of one or more of his computers. Self learning may be the easiest way a user, not familiar with the inner workings of the system, can update the rule base to adjust to these changes.

So far, we have considered all our examples as capable of dealing with yes/no responses only. What happens if the user doesn't know the answer? And what happens if the response cannot be as simple as yes or no? The mathematics of our experts systems get a little more complex as we consider these items in the next part of this series.

Fig 4. Example of Rule Learning/Modification

```
100 REMark Simple Expert System
110 REMark  Learning new rules
120 REMark    by iteration
130 REMark
140 REMark David Nowotnik
150 REMark February, 1987
160 REMark
170 REMark Initialise
180 DIM f$(7,30): DIM a$(5,14): DIM r(5,7): DIM s(5,7): DIM
y(5)
190 RESTORE
200 REMark Features
210 DATA "128K of RAM"
220 DATA "Large selection of games"
230 DATA "Z80 processor"
240 DATA "Built-in cassette"
250 DATA "Built-in RS232 port"
260 DATA "Comes with business programs"
270 DATA "Proper keyboard"
280 REMark Answers
290 DATA "Spectrum 48K",0,1,1,0,0,0,0
300 DATA "Spectrum Plus",0,1,1,0,0,0,1
310 DATA "Spectrum 128",1,1,1,0,0,0,1
320 DATA "Spectrum +2",1,1,1,1,0,0,1
330 DATA "QL",1,0,0,0,1,1,1
340 FOR i=1 TO 7
350 READ b$: LET f$(i)=b$
360 NEXT i
370 FOR i=1 TO 5
380 READ b$: LET a$(i)=b$
390 FOR j=1 TO 7
400 READ c: LET r(i,j)=c
410 LET s(i,j)=0
420 NEXT j
430 NEXT i
440 :
450 CLS
460 PRINT " Expert System - Learning Rules"
470 PRINT: PRINT: PRINT " Select:": PRINT
480 PRINT " 1. Test the knowledge base"
490 PRINT " 2. Enter profiles "
500 PRINT " 3. Auto-learning"
510 PRINT "      ";: INPUT d$
520 IF d$="1" THEN PRINT : GO TO 560
530 IF d$="2" THEN GO TO 640
540 IF d$="3" THEN GO TO 760
550 GO TO 510
560 REMark Test the knowledge base
570 FOR k=1 TO 5
580 LET i=k: GO SUB 2120: GO SUB 2200
590 PRINT: PRINT "My answer is ";a$(high)
600 PRINT: PRINT " Press ENTER to continue"
610 INPUT d$
620 NEXT k
630 GO TO 450
640 REMark Manual entry of Profiles
650 FOR h=1 TO 5
```

```
660 CLS
670 GO SUB 2000
680 GO SUB 2200
690 GO SUB 2400
700 GO SUB 2700
710 GO SUB 3000
720 PRINT: PRINT " Press ENTER to continue"
730 INPUT d$
740 NEXT h
750 GO TO 450
760 REMark Auto select profiles
770 FOR h=1 TO 5
780 CLS: PRINT "Auto Select Profiles": PRINT
790 GO SUB 2100
800 GO SUB 2200
810 GO SUB 2600
820 IF ok THEN GO TO 860
830 GO SUB 2600
840 GO SUB 2700
850 GO SUB 3000
860 PRINT: PRINT " Press ENTER to continue"
870 INPUT d$
880 NEXT h
890 GO TO 450
2000 REMark Enter profiles
2010 LET t$="": PRINT "Enter y or n to each question": PRINT
2020 FOR i=1 TO 7
2030 PRINT f$(i);" ";
2040 INPUT d$: IF d$="" THEN GO TO 2040
2050 IF d$(1)="Y" OR d$(1)="y" THEN LET t$=t$&"1": GO TO 2080
2060 IF d$(1)="N" OR d$(1)="n" THEN LET t$=t$&"0": GO TO 2080
2070 GO TO 2040
2080 NEXT i
2090 RETurn
2100 REMark Auto select profile
2110 LET i= 1+RND(4)     [ LET i= 1+ INT (RND *5) on Spectrum]
2120 PRINT: PRINT a$(i);" selected."
2130 LET sl=i: LET t$="0000000"
2140 FOR j=1 TO 7
2150 IF r(sl,j)="1" THEN LET t$(j)="1"
2160 NEXT j
2170 RETurn
2200 REMark Guess!
2210 LET tot=0: LET high=1
2220 FOR i=1 TO 5
2230 LET q=0
2240 FOR j=1 TO 7
2260 IF t$(j)="1" THEN LET q=q+s(i,j)
2270 NEXT j
2280 LET y(i)=q: IF q>tot THEN LET tot=q: LET high=i
2290 NEXT i
2300 RETurn
2400 REMark Manual Correct
2410 PRINT: PRINT "My answer is ";a$(high)
2420 PRINT " Is that right? (y/n)  ";
2430 INPUT d$: IF d$="" THEN GO TO 2430
2440 IF d$(1)="Y" OR d$(1)="y" THEN LET ok=1: RETurn
2450 IF d$(1)="N" OR d$(1)="n" THEN GO TO 2470
2460 GO TO 2430
2470 LET ok=0
2480 PRINT : PRINT "Which one should it be? ": PRINT
2490 FOR i=1 TO 5
2500 PRINT i;" ";a$(i)
2510 NEXT i
2520 PRINT :PRINT "Which one is it? (1-5) ";
2530 INPUT d$: IF d$="" THEN GO TO 2530
2540 IF d$(1)<"1" OR d$(1)>"5" THEN GO TO 2530
2550 LET rt= CODE(d$)-48
2560 RETurn
2600 REMark Auto correct
2610 PRINT: PRINT "My answer is ";a$(high);
2620 IF high<>sl THEN GO TO 2650
2630 LET ok=1: PRINT ". That's right"
2640 RETurn
2650 LET ok=0: PRINT ". That's wrong"
2660 PRINT "I'm updating the rules"
2670 LET rt=sl
2680 RETurn
2700 REMark Update rules
2710 FOR i=1 TO 5
2720 IF i=rt OR y(i)<tot THEN GO TO 2760
2730 FOR j=1 TO 7
2740 IF t$(j)="1" THEN LET s(i,j)=s(i,j)-1
2750 NEXT j
2760 NEXT i
2770 FOR j=1 TO 7
2780 IF t$(j)="1" THEN LET s(rt,j)=s(rt,j)+1
2790 NEXT j
2800 RETurn
3000 REMark Print New Rules
3010 FOR i=1 TO 5
3020 LET x$=a$(i)&"              ": LET x$=x$(1 TO 14): PRINT
x$;":"
3030 FOR j=1 TO 7
3040 IF s(i,j)>=0 THEN PRINT " ";
3050 PRINT s(i,j);" ";
3070 NEXT j
3080 PRINT
3090 NEXT i
3100 RETurn
```

Please note in line 2110; the line as written is the QL version; for the Spectrum, substitute the command in brackets (and bold print) in the listing.

CROSSFIRE

Views on products old and new in this month's post bag

Beta Plus

Could you please print an occasional article on the Beta Plus Interface? Since Ray Elder's excellent review (Dec/Jan, 85/86) I am sure many readers must have bought this system. However I haven't heard any more about this interface until your review of the Hisoft Compiler mentioned that the compiler would not work with the Beta Interface. Even mentioning utilities which can be converted to work with Beta would be helpful.

The people at Print 'N Plotter (01-403-3622) have been very helpful, and I found their Paint Plus art utility converted easily to work with the Beta Plus and is well worth doing.

Great magazine, could be perfect! No threats, I'll keep on buying it.
Cyril Heywood,
London.

Some popular pieces of hardware, such as the Discovery disc drive, give rise to their own user groups which are often the main source of information for interested users, but as far as we know there aren't any groups for the Beta system so there's not much being written about it. Maybe there's someone out there who runs such a group and would like to get in touch with us?

'81 no more

Having read about P.C. Allen's request for ZX81 hardware, I thought I'd drop you a line about the "Pine Marten" newsletter. I've enought replies to send out the second issue of the newsletter — and PM will include an offer for a joystick interface adaptor. The adaptor will enable Spectrum joystick interfaces to be used on the ZX81. The newsletter is totally free to contributors (ie. almost anyone who writes in to it). There is now a charge of 40p for the newsletter for those who can't think of anything to say. Overseas readers may send two IR coupons.

Since last writing to you I have decided not to produce any more software for the ZX81; the HiRes utility may still be released, work on it has been given very low priority so I can't say when it might be finished. I've sold only around 20 copies of "War Web" to date and this doesn't justify the cost of the mailers (96 on the mailing list) or the advertising.

I will no longer be advertising ZX81 games, but I would like to reassure your readers that I will continue to supply the current range of games for the ZX81 and "Pine Marten" will continue as long as people write in.
Gary Rowland (Pooter Games),
Dagenham, Essex.

Hi, Beta!

I was very interested in the review of the HiSoft Compiler in the February 1987 issue which arrived today. I bought the programme to speed up the calculations in the Multiple Linear Regression program from University Software which I use.

I think some of the points I found may be of interest to your readers. I tried to compile only the calculation section as the manual said one could move in and out of BASIC, but this does not apply if the parts left in BASIC use variables from the compiled section.

The original program used variables in DIM statements. I had to decide on maxima for these to use numbers in the DIM statements.

The other point is not to use byte sving ersions for the compiling. By replacing **PI** *byte savers and* **VAL"number"** *with actual numbers the compiled code used about 1.5K less on my printer version of the program. The compiler version uses more bytes than the BASIC one as noted by your anonymous reviewer but the speed increase is worth this. With the BASIC version result calculation could take up to fifteen minutes with ten independent variables — the compiled version takes 30 seconds.*

I have found the option of a Microdrive backup very useful.
J.F. Osborne,
Huntingdon, Cambs.

the DISCIPLE

COMPETITION

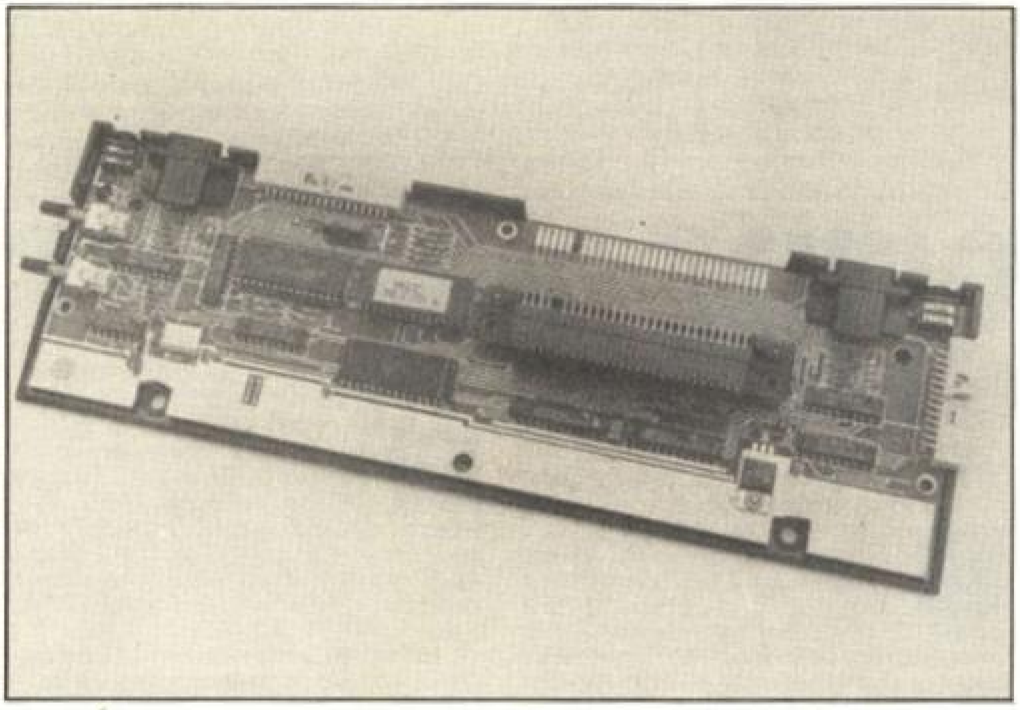
Your chance to win the most exciting Spectrum add-on to appear for a very long time

The Disciple is an immensely versatile multi-purpose interface that functions as a disk, printer, dual joystick and network interface for all computers in the Spectrum range.

Cunningly designed to resemble the Interface 1, the Disciple uses the same raised connector leaving the Spectrum's input and output ports free to use. With features such as snapshot button allowing up to 16 snapshot files to be saved when working with double density, the Disciple has found favour among serious Spectrum users since its recent introduction.

We are offering two Disciples in this competition. All you have to do is select the right answers to the questions below and ring them on the entry coupon. Clues to the answers can be found in John Wase's Disciple review in last month's ZX and perhaps on this page. Or, to avoid educated guesswork perhaps Rockfort Products will send you a brochure if you ring them on the number to be found elsewhere in this issue.

The first two correct entries picked out of the hat will receive a Disciple.



The Questions

1. How many 48K commercial games could save on a double sided double density disc using the Disciple with an 80 track disk drive?
a) 8.
b) 12.
c) 16.

2. Loading a file from disc with the Disciple is approximately:
a) 20 times
b) 40 times
c) 60 times
faster than loading the same file from tape.

3. To copy a screen from a commercial program to your printer with the Disciple, you simply:
a) Press Cap Shift and the snapshot button.
b) Enter LPRINT.
c) Press P and Snapshot button.

4. Which statement is incorrect?
a) The Disciple uses hook codes similar to Interface 1.
b) To transfer most commercial programs from tape or microdrives to disc press the snapshot button.
c) The joystick ports are Kempston, Interface II and Protek.

The competition is open to all ZX readers except employees of Argus Specialist Publications, Chase Web, Rockfort Products, and Miles Gordon Technology. The editors decision is final and no correspondence can be entered into.

Send your entries to: Disciple Competition, ZX Computing Monthly, No 1 Golden Square, London W1R 3AB.

Please remember to put your address on your entry envelope.

The closing date for entries is May 1st 1987.

Disciple Competition

Please ring the appropriate letter

1.	A	B	C
2.	A	B	C
3.	A	B	C
4.	A	B	C

Name:..

Address: ....

....

Send your entries to Disciple Competition, ZX Computing, No 1 Golden Square, London W1R 3AB

ELEVATOR ACTION



A game of ups and downs from Quicksilva

Elevator Action
Quicksilva
£8.95

After a series of pretty unexciting titles it looks like Quicksilva are finally starting to show signs of life again. Their recent Glider Rider wasn't earth-shattering but it did have some nice touches, and now Quicksilva have joined the rest of the industry by turning to coversions of successful arcade games.

I've never played the original coin-op version of Elevator Action so I don't know how faithful a conversion this is, but after a slow start it did turn out to be simple, if undemanding, fun.

The game is set inside a tall building where all the floors are interconnected, both horizontally and vertically, by numerous lifts and escalators. On each floor there are a number of doors, painted either red or yellow, and behind the red doors are secret papers that you have to collect before you can leave the building through the basement. You enter through the roof and head downwards, though you might find that you have to do a bit of shuttling up and down as you go after all the papers.

Unlike our own secret service chaps who go up to complete strangers in the street and offer them bags full of official secrets, the owners of these papers have sensibly arranged for them to be protected by armed guards who prowl each floor and come popping out of the locked, yellow doors at the least convenient moment. When you come face to face with the guards you can either run for the nearest lift, or, if there isn't one around you can kick or fire at them with your own gun. The trouble here is that the guards often manage to get off a few shots of their own before they bite the dust, but if your timing's up to it you can jump or duck out of the line of fire.

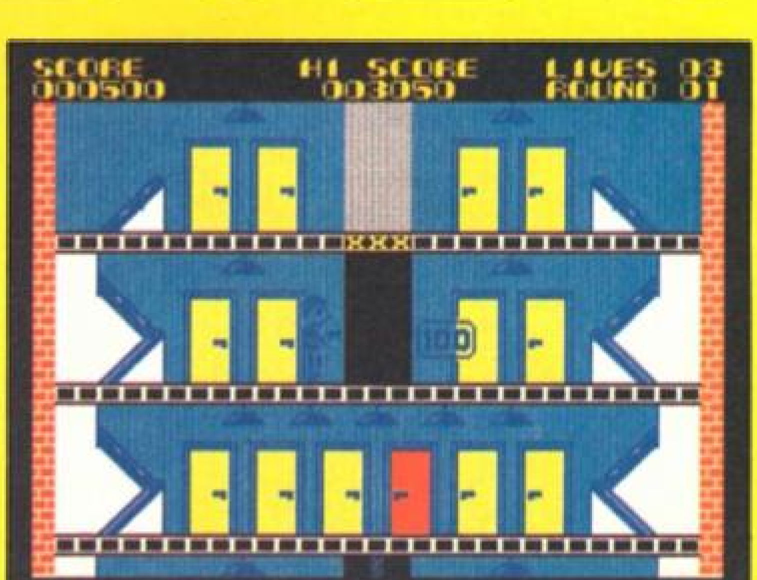
At the top of the building there's just the one lift, and no guards to avoid, but the further down you go the more complicated the interconnections become and the more guards there are wandering around. Fortunately, once you're in a lift you can control its movements with the joystick, and this allows you to go straight to the areas with the red doors, rather than having to shoot your way through each floor individually.

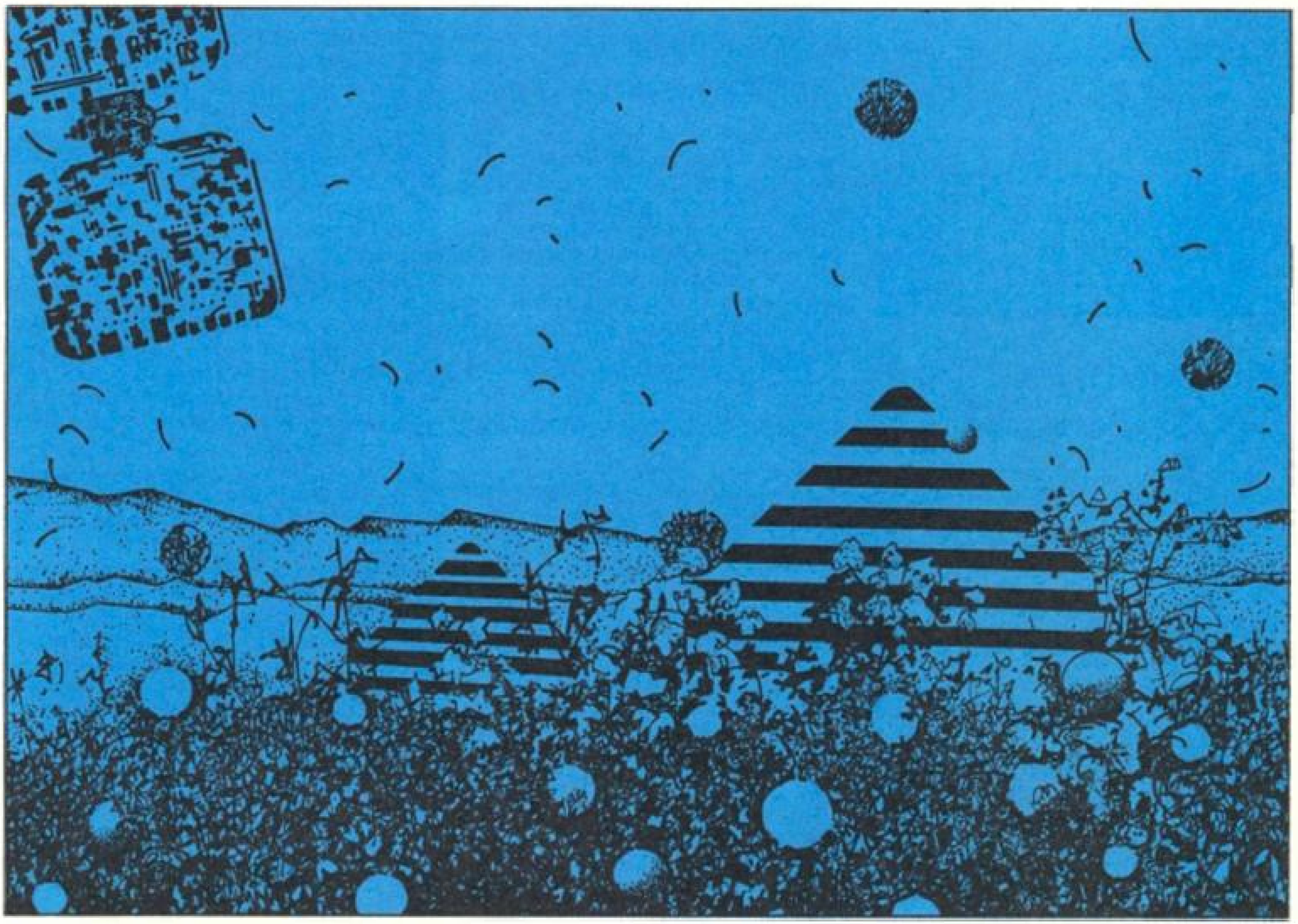
For some reason, when you get to the exit at the bottom of the building, there are some weird colour clashes, but these don't affect the game itself and you then go on to the start of the next game.

It's not the fastest, most action packed arcade game I've ever come across but for some reason it does seem to be a bit addictive (possibly because it is so simple for once — it makes a change not to have to read a great wodge of instructions before getting into the game). It would be even beter if the layout of the building changed at the end of each section, instead of using the same building with more doors and guards — as it stands I think the game's life is probably limited by the number of times that you're prepared to go over the same ground. The game is also a bit overpriced at £8.95, but if you want your collection of arcade conversions to be complete then Elevator Action is worth a look.









RANDOM LANDSCAPE GRAPHICS

Alan Davis presents a technique for creating random adventure backgrounds

Despite the arrival of the 128K Spectrum, many of us are still trundling along with our old 48K machines, and are generally fairly content I suspect. Speaking purely personally, and confining my remarks to the adventure game genre, I don't believe that the potential of 48K has been exhausted yet (or anywhere near it). But having said that, I must admit that there's one aspect of adventure programming where the limitations of 48K are very evident. I'm talking about graphics, of course. They gobble memory mercilessly, and any adventure writer is always on the look-out for programming methods which can help to selve this problem.

One possible solution is to use some method of "streamlining" the production of at least part of each garphic illustration, perhaps by using a small number of subroutines to set up a limited number of backgrounds onto which different shapes can be superimposed. In this way you could arrange for each location in your adventure to have some sort of illustration — though there's a price to pay of course. Inevitably there'd be a certain sameness about them, even though they might differ in detail.

One program which used something like this technique was the immensely successful "Valhalla", which has been given a new lease of life recently by being released as a budget game. Again, speaking purely for myself, I always felt that the game itself was slightly less interesting than cold rice pudding, but I must admit that I do find the illustrations highly atmospheric. So the method clearly can work, and yet oddly enough nothing quite like it seems to have been attempted since. Shall we have a go?

Background work

I don't propose to do more in this article than to tackle the job of drawing backgrounds, because this alone generated quite enough interesting little problems to keep us occupied for a month! At the outset, I decided it would be a rather neat idea to let the Spectrum's random generator take over the creation of the picture, and began (as I so often do) by trying out a few simple ideas in BASIC. **Listing 1** will show you the sort of thing I was aiming for, and if you type it in and run it you'll get some idea of the effects that can be achieved even with these simple routines (It's ridiculously slow of course, but that doesn't matter for the moment). The screen shot which should be lurking somewhere about these pages shows an application of the same principles, extended to use the entire screen, and with some graphics for buildings superimposed to show how the ideas could be used in practice.

As you'll see, this little

program uses the RND function to determine the operation of the various PLOT and DRAW routines, and it's capable of producing some rather effective mountains, rippled water, reeds, and so on. But what makes the system of potential interest to the adventure programmer is the fact that, as you know, the RND function doesn't actually produce random numbers at all; it produces pseudo-random numbers. This means that we get a strictly predetermined sequence, which repeats itself only after 65536 numbers have been generated. The starting point of the sequence is set by using RANDOMIZE n, and you'll see this used at line 1000. Every time you press a key in response to the "Next picture?" prompt, the value of n is increased so that the sequence begins next time from a different point and produces a different picture. The changes are most noticeable in the mountain profiles mainly because these are the most complex shapes (being the ones I spent the most time tinkering with).

I think you'll see now what I'm getting at. If we let n represent the "location number" in our adventure, and precede every call to the graphics subroutines by "RANDOMIZE n", we'll always get the same set of background graphics for the same location, yet the background for each location will be different. Of course you could go much further than I've done here. Other subroutines can be added (and called conditionally on the value of n). Just by changing the ink and paper the lake in the middle distance can be made to turn into a marsh, an ice sheet, a sandy desert, or whatever.

Speed trap

Unfortunately we still have the problem of speed — or rather, lack of it! Aha, you say. No problem. PLOT, DRAW, and RND can easily be coped with in machine code by making suitable calls to the ROM. Half an hour's work with an assembler, and we're home and dry. Well, don't let me stop you, but I'm afraid you'll find the exercise very disappointing. The chief snag (there always is one, isn't there?) is that the normal Spectrum RND routine is horrendously slow, and using it from machine code doesn't actually make much difference. We'd better think again.

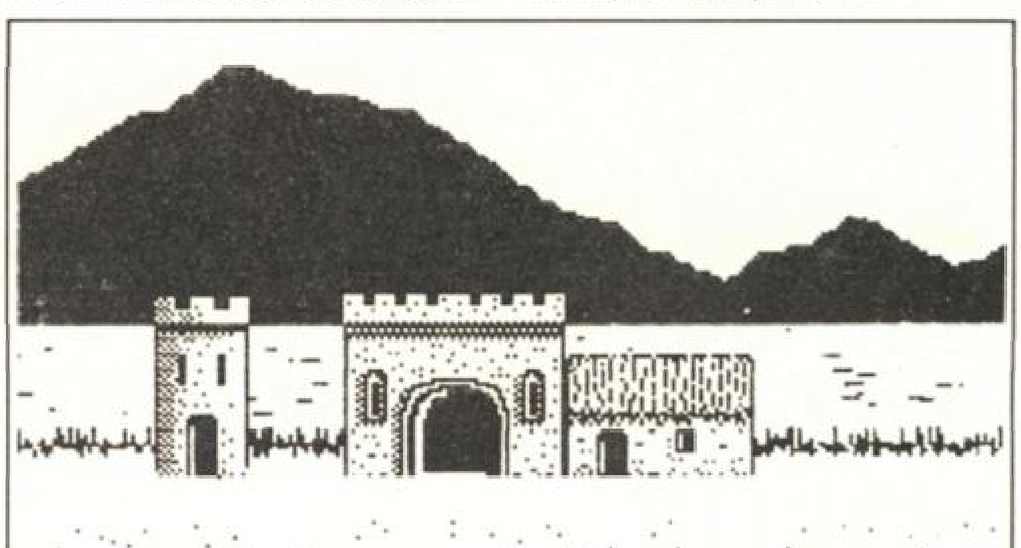
Strictly speaking, we don't actually need all the possible permutations in the RND sequence for the job in hand. Nor do we need 65536 different pictures, come to that. Wouldn't 256 be enough? Well then, why don't we simply stash away in memory a table of 256 pseudo-random integers, and then extract our random numbers from the tables as and when they're needed?

Let me explain in a little more detail. First, we reserve 256 bytes of memory (in fact I used addresses 64744 to 64999 inclusive) and fill each of those bytes with a random integer like this:

FOR I = 64744 TO 64999: POKE I, INT(RND * 255): NEXT I

Listing 1

```
   1 DEF FN r(x)=INT (RND*x)
   2 BORDER 0: INK 0: LET n=1: GO TO 1000
   7 REM
   8 REM **DRAW RECTANGLE AND FILL WITH BACKGROUND PAPER**
   9 REM
  10 CLS : PLOT 0,95: DRAW 128,0: DRAW 0,80: PRINT AT 1,17;"PICT
URE NUMBER";AT 3,24;n;AT 0,0;
  15 PRINT  PAPER 5,',',',',' PAPER 1,',' PAPER 6,',',': RETURN
  17 REM
  18 REM **DRAW MOUNTAINS**
  19 REM
  20 INK 0: LET ht=FN r(24): LET mx=23: LET mn=0: LET p=FN r(2)
  25 FOR i=0 TO 127: PLOT i,136: DRAW 0,7+ht
  30 IF p THEN  LET ht=ht+FN r(2): IF ht>=mx THEN  LET p=0: LET
ht=mx: LET mn=1+FN r(7): GO TO 50
  40 IF NOT p THEN  LET ht=ht-FN r(2): IF ht<=mn THEN  LET p=1:
LET ht=mn: LET mx=8+FN r(16)
  50 NEXT i: RETURN
  77 REM
  78 REM **DRAW LAKE**
  79 REM
  80 INK 7: FOR i=0 TO 49
  90 LET x=FN r(125): LET y=128+FN r(8): PLOT x,y
 100 DRAW FN r(4),0: NEXT i: RETURN
 137 REM
 138 REM **DRAW REEDS**
 139 REM
 140 INK 4: FOR i=0 TO 127: LET mn=FN r(2): LET mx=1+FN r(7)
 150 PLOT i,119+mn: DRAW 0,mx-mn: NEXT i: RETURN
 177 REM
 178 REM **DRAW ROUGH GROUND SPECKLE**
 179 REM
 180 INK 0: FOR i=0 TO 49
 190 LET x=FN r(128): LET y=104+FN r(8): PLOT x,y: NEXT i: RETUR
N
 237 REM
 238 REM **DRAW FOREGROUND GRASS**
 239 REM
 240 INK 4: FOR i=0 TO 127: PLOT i,96: DRAW 0,FN r(8): NEXT i
 250 RETURN
 997 REM
 998 REM **MAIN LOOP FOR DRAWING COMPLETE PICTURE**
 999 REM
1000 CLS : RANDOMIZE n: GO SUB 10: REM ** CLEAR RECTANGLE
1005 GO SUB 20: REM ** DRAW MOUNTAINS
1010 GO SUB 80: REM ** DRAW LAKE
1015 GO SUB 140: REM ** DRAW REEDS
1020 GO SUB 180: REM ** DRAW GROUND
1025 GO SUB 240: REM ** DRAW GRASS
1030 INK 0: PRINT #1;AT 1,9; FLASH 1;"NEXT PICTURE?"
1040 PAUSE 0: LET n=n+1: GO TO 1000
```



Just one landscape effect achieved with this utility

Now imagine that we have a pointer which can be moved along the table. Whenever we want a random number we just collect the entry in the table indicated by the current position of the pointer and move the pointer along one notch ready for the next random number collection (When the pointer gets to the end of the table, we simply arrange for it to move back to the start again). This way we can get 256 different pictures according to the initial position of the pointer. Of course you'll rightly point out that during the drawing of one complete picture the pointer will cycle many times through the table, repeating itself every time. Indeed it will — but oddly enough it doesn't matter much in practice. A hint of regularity in things like the "grass" or "reeds" graphics will hardly be obtrusive anyway and with more complicated shapes like mountains the use of the random numbers by the graphic routines is complex enough to mask almost completely the underlying repetitions. The techniques wouldn't suit a statistician — but it'll do for us.

Translation

We're now in a postiton to attempt what we set out to do — that is, to "translate" the slow bits of our original BASIC program into machine code. In order to do so we need 4 essential subroutines, and these are given in **Listing 2**. The first two are just the well-known methods of using the ROM routines for drawing a line and plotting a point. To get the equivalent of PLOT x,y we load the C register with "x", the B register with 'y", and then CALL PLOT. To get the equivalent of DRAW x,y we load the B and C registers similarly and then CALL DRAW.

The interesting stuff comes with the third routine (labelled RAND in Listing 2). I've annotated this in some detail so that all I need to do here is explain how it's used. It produces the equivalent of INT (RND*x) using a table of 256 random integers as we discussed above. The routine is entered with the A register holding the value of "x", and on return you'll find the A register holding INT (RND*x). In other words, it does much the same job as the defined function in Listing 1. You'll need to fill the table from BASIC yourself of course as I described earlier — but once you've done that the table can be saved along with the rest of the code as a single block. You control the starting point of the RAND sequence by POKEing 65057 (POINT) with the current location number in your adventure — this is effectively the "pointer" we were talking about before.

Listing 2

```
*HISOFT GENS3M2 ASSEMBLER*
      ZX SPECTRUM

Copyright (C) HISOFT 1983,4
All rights reserved

Pass 1 errors: 00

           *C-
           ;DRAW, PLOT, and "RND" routines
           *D+
65000              ORG  65000
           DRAW
           ;This is the equivalent of DRAW C,B
65000              EXX
65001              PUSH HL
65002              EXX
65003              LD   DE,#0101
65006              CALL #24BA
65009              EXX
65010              POP  HL
65011              EXX
65012              RET



           ;This is equivalent to PLOT C,B
 8933      PLOT    EQU  #22E5


           RAND
           ;This replaces contents of A register by INT(RND*A)
           ; Note that addresses 64744 to 64999 must contain
           ; random integers between 0 and 254 inclusive
65013              LD   (VALUE),A; Store A in (VALUE)
65016              LD   HL,64744; Start of random number table
65019              LD   DE,(POINT); Current value of pointer
65023              LD   D,0
65025              ADD  HL,DE
65026              LD   A,E
65027              INC  A
65028              LD   (POINT),A; Store next value of pointer
65031              LD   A,(HL); Pick up random integer (r)
65032              CALL #2D28; Stack it
65035              LD   A,255
65037              CALL #2D28; Put 255 on stack
65040              RST  40; Calculator on
65041              DEFB 5; Divide to leave "RND" on stack
65042              DEFB 56; Calculator off
65043              LD   A,(VALUE)
65046              CALL #2D28; Put (VALUE) on stack
65049              RST  40; Calculator on
65050              DEFB 4; Multiply (VALUE) by "RND"
65051              DEFB 39; Effectively INT(VALUE*"RND")
65052              DEFB 56; Calculator off
65053              CALL #2DD5; Result to A register
65056              RET
65057      POINT   DEFB 0
65058      VALUE   DEFB 0


           BKGRND
           ; This draws the picture frame, and then prints
           ; horizontal stripes of background paper
65059              LD   A,2
65061              CALL #1601; Select screen for printing
65064              LD   B,95
65066              LD   C,0
65068              CALL PLOT
65071              LD   B,0
65073              LD   C,128
65075              CALL DRAW
65078              LD   B,80
65080              LD   C,0
65082              CALL DRAW
65085              LD   A,17
65087              RST  16
65088              LD   A,(PAP1)
65091              RST  16
65092              LD   B,5; Five rows of paper (PAP1)
65094      LOOP1   LD   A,6
65096              RST  16
65097              LD   A,13
65099              RST  16
65100              DJNZ LOOP1
65102              LD   A,17
65104              RST  16
65105              LD   A,(PAP2)
65108              RST  16
65109              LD   B,2; Two rows of paper (PAP2)
```

```
65111  830 LOOP2  LD   A,6
65113  840        RST  16
65114  850        LD   A,13
65116  860        RST  16
65117  870        DJNZ LOOP2
65119  880        LD   A,17
65121  890        RST  16
65122  900        LD   A,(PAP3)
65125  910        RST  16
65126  920        LD   B,3; Three rows of paper (PAP3)
65128  930 LOOP3  LD   A,6
65130  940        RST  16
65131  950        LD   A,13
65133  960        RST  16
65134  970        DJNZ LOOP3
65136  980        RET
65137  990 PAP1   DEFB 5; Cyan
65138 1000 PAP2   DEFB 1; Blue
65139 1010 PAP3   DEFB 6; Yellow

Pass 2 errors: 00

Table used:    152  from   331
```

The final subroutine is the one labelled BKGRND. This is completely self-contained, and just clears out a rectangle on the screen, filling it with suitably coloured strips of paper, and drawing a line around it. In fact it does mainly the same job as lines 10/15 in Listing 1. There are just two points to make here. First, I've arranged for the picture to take up roughly one quarter of the screen, so as to leave plenty of space for text. Secondly, you'll probably want to fiddle with the paper colours at some stage, either from within a program or just to try out different effects — and you can do this by poking different values into 65137/8/9 before calling the routine. For

Listing 3

```
*HISOFT GENS3M2 ASSEMBLER*
       ZX SPECTRUM

Copyright (C) HISOFT 1983,4
All rights reserved

Pass 1 errors: 00

         10 *C-
         20 ;Picture components
         30 *D+
65140    40        ORG  65140
       1020
       1030 MNTNS
65140 1040        CALL INIT
65143 1050        LD   A,24
65145 1060        CALL RAND
65148 1070        LD   (HT),A
65151 1080        LD   A,23
65153 1090        LD   (MX),A
65156 1100        XOR  A
65157 1110        LD   (MN),A
65160 1120        LD   A,2
65162 1130        CALL RAND
65165 1140        LD   (PP),A
65168 1150 LOOP4  LD   A,(II)
65171 1160        LD   C,A
65172 1170        LD   B,136
65174 1180        CALL PLOT
65177 1190        XOR  A
65178 1200        LD   C,A
65179 1210        LD   A,(HT)
65182 1220        ADD  A,7
65184 1230        LD   B,A
65185 1240        CALL DRAW
65188 1250        LD   A,(PP)
65191 1260        CP   0
65193 1270        JP   NZ,UP
65196 1280        LD   A,2
65198 1290        CALL RAND
65201 1300        LD   B,A
65202 1310        LD   A,(HT)
65205 1320        SUB  B
65206 1330        LD   (HT),A
65209 1340        LD   B,A
65210 1350        LD   A,(MN)
65213 1360        CP   B
65214 1370        CALL NC,SWAP2
65217 1380 CONT   LD   A,(II)
65220 1390        INC  A
65221 1400        CP   128
65223 1410        RET  Z
65224 1420        LD   (II),A
65227 1430        JP   LOOP4
65230 1440 UP     LD   A,2
65232 1450        CALL RAND
65235 1460        LD   B,A
65236 1470        LD   A,(HT)
65239 1480        ADD  A,B
65240 1490        LD   (HT),A
65243 1500        LD   A,(MX)
65246 1510        LD   B,A
65247 1520        LD   A,(HT)
65250 1530        CP   B
65251 1540        CALL NC,SWAP1
65254 1550        JP   CONT
65257 1560 SWAP1  LD   A,B
65258 1570        LD   (HT),A
65261 1580        LD   A,7
65263 1590        CALL RAND
65266 1600        ADD  A,1
65268 1610        LD   (MN),A
65271 1620        XOR  A
65272 1630        LD   (PP),A
65275 1640        RET
65276 1650 SWAP2  LD   (HT),A
65279 1660        LD   A,16
65281 1670        CALL RAND
65284 1680        ADD  A,8
65286 1690        LD   (MX),A
65289 1700        LD   A,1
65291 1710        LD   (PP),A
65294 1720        RET
65295 1730 II     DEFB 0
65296 1740 PP     DEFB 0
65297 1750 HT     DEFB 0
65298 1760 MX     DEFB 0
65299 1770 MN     DEFB 0
       1780
       1790
       1800 LAKE
65300 1810        CALL INIT
65303 1820 LOOP5  LD   A,125
65305 1830        CALL RAND
65308 1840        LD   C,A
65309 1850        LD   A,8
65311 1860        PUSH BC
65312 1870        CALL RAND
65315 1880        POP  BC
65316 1890        ADD  A,128
65318 1900        LD   B,A
65319 1910        CALL PLOT
65322 1920        LD   A,4
65324 1930        CALL RAND
65327 1940        LD   C,A
65328 1950        LD   B,0
65330 1960        CALL DRAW
65333 1970        LD   A,(II)
65336 1980        INC  A
65337 1990        CP   50
65339 2000        RET  Z
65340 2010        LD   (II),A
65343 2020        JP   LOOP5
       2030
       2040
       2050 REEDS
65346 2060        CALL INIT
65349 2070 LOOP6  LD   A,2
65351 2080        CALL RAND
65354 2090        LD   (MN),A
65357 2100        LD   A,7
65359 2110        CALL RAND
65362 2120        ADD  A,1
65364 2130        LD   (MX),A
65367 2140        LD   A,(II)
65370 2150        LD   C,A
65371 2160        LD   A,(MN)
65374 2170        ADD  A,119
65376 2180        LD   B,A
65377 2190        CALL PLOT
65380 2200        XOR  A
65381 2210        LD   C,A
65382 2220        LD   A,(MN)
65385 2230        LD   B,A
65386 2240        LD   A,(MX)
65389 2250        SUB  B
65390 2260        LD   B,A
65391 2270        CALL DRAW
65394 2280        LD   A,(II)
65397 2290        INC  A
65398 2300        CP   128
65400 2310        RET  Z
65401 2320        LD   (II),A
65404 2330        JP   LOOP6
       2340
       2350
       2360 GROUND
65407 2370        CALL INIT
65410 2380 LOOP7  LD   A,128
65412 2390        CALL RAND
65415 2400        LD   C,A
65416 2410        LD   A,8
65418 2420        PUSH BC
65419 2430        CALL RAND
65422 2440        POP  BC
65423 2450        ADD  A,104
65425 2460        LD   B,A
65426 2470        CALL PLOT
65429 2480        LD   A,(II)
65432 2490        INC  A
65433 2500        CP   50
65435 2510        RET  Z
65436 2520        LD   (II),A
65439 2530        JP   LOOP7
       2540
       2550
       2560 GRASS
65442 2570        CALL INIT
65445 2580 LOOP8  LD   A,(II)
65448 2590        LD   C,A
65449 2600        LD   B,96
65451 2610        CALL PLOT
65454 2620        LD   A,8
65456 2630        CALL RAND
65459 2640        LD   B,A
65460 2650        LD   C,0
65462 2660        CALL DRAW
65465 2670        LD   A,(II)
65468 2680        INC  A
65469 2690        CP   128
65471 2700        RET  Z
65472 2710        LD   (II),A
65475 2720        JP   LOOP8
       2730
       2740
       2750 INIT
       2760 ;Transparent paper
65478 2770        LD   A,248
65480 2780        LD   (MASKT)
       2790 ;Set counter to zer
65483 2800        XOR  A
65484 2810        LD   (II),A
65487 2820        RET
23696 2830 MASKT  EQU  23696
 8933 2840 PLOT   EQU  8933
65000 2850 DRAW   EQU  65000

65013 2860 RAND   EQU  65013

Pass 2 errors: 00

Table used:    278  from   367
```

example, if you want your picture to appear as though night had fallen you could POKE 65137,1. This would change the paper for the top 5 strips (the "sky") from cyan to blue. To turn the "lake" into a "marsh" you might POKE 65138,4 — and so on.

All that now remains to be done is to "translate" each of the separate graphics routines into its assembly language equivalent, and I've done this in **Listing 3**. Although this isn't annotated (if it was it would take up far more space than it's worth) I've stuck as closely as I can to the BASIC in Listing 1 so that you should be able to follow it without too much difficulty. In any case you'll almost certainly want to tinker with these, adding your own extra routines and possibly making the existing ones look more interesting on screen. If you want to run the whole thing just as it stands, then simply assemble Listing 3 at the end of Listing 2, leaving out the ORG 65140 instruction and the PLOT, DRAW, and RAND label definitions at the end.

Picture demo

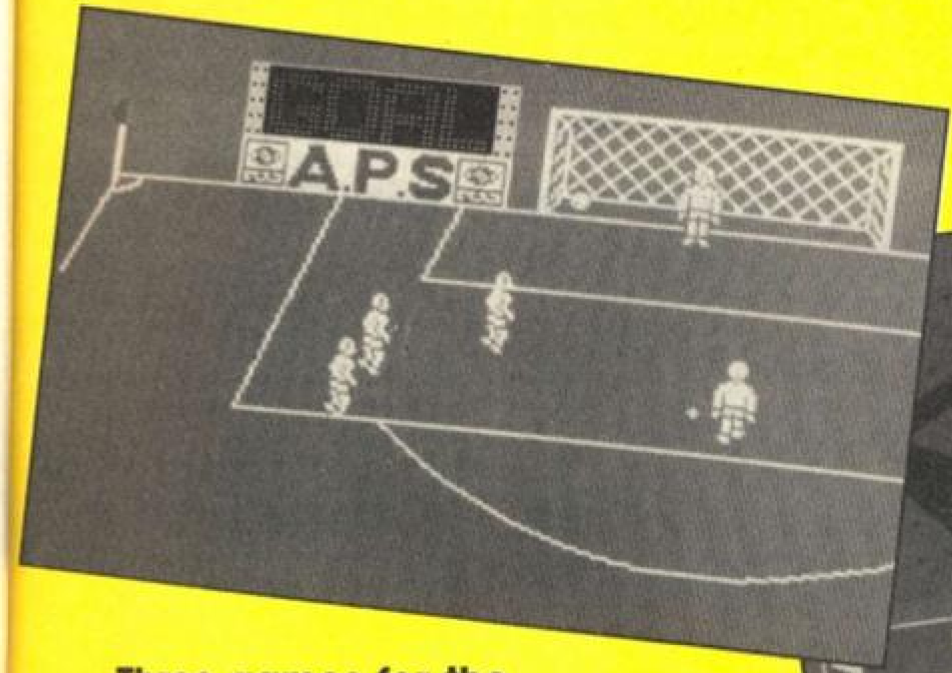
Once you've assembled the code and saved it (and don't forget that RND table, whatever you do!), you'll need a little BASIC program to drive it. **Listing 4** will do a demonstration job, cycling through the 256 pictures as you press a key. You'll find that production of the pictures is now quite fast enough for the system to be useful — and very considerably faster, in fact, than we'd have managed if we'd stuck with the RND routine normally used. The price of this improvement in speed is just the 256 bytes needed to store the random number table, which I think you'll agree is a very acceptable bargain.

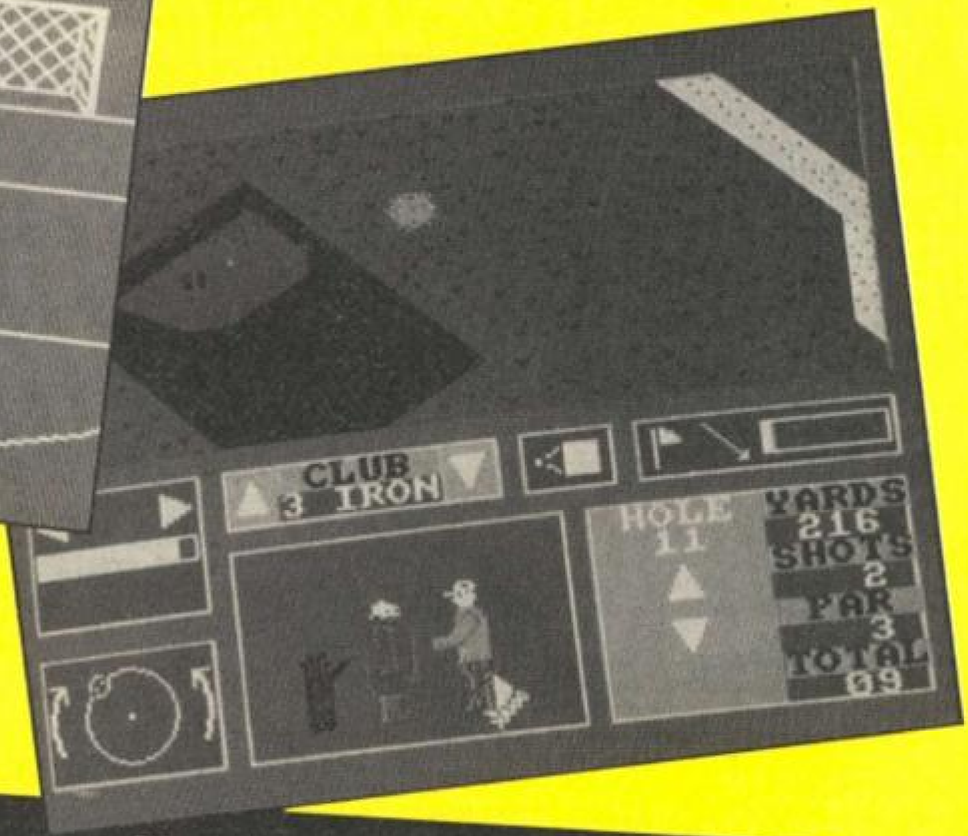
As you'll see, the graphics routines themselves take up very little memory, and you can afford to add to these without much anxiety. Of course these backgrounds won't do on their own, but arranging a few sub-routines to print up the shapes of castles, huts, and the like, in blocks of character squares isn't difficult. I have a little designer program to deal with this, which stores the graphics for buildings in an alternative character set so that they can be printed up using RST 16. But that might be a job for some future article, and in the meantime you'll doubtless have your own ideas about ways of using the random back-grounds. Coulds yours be the first adventure to use "RANDSCAPING", perhaps . . .?

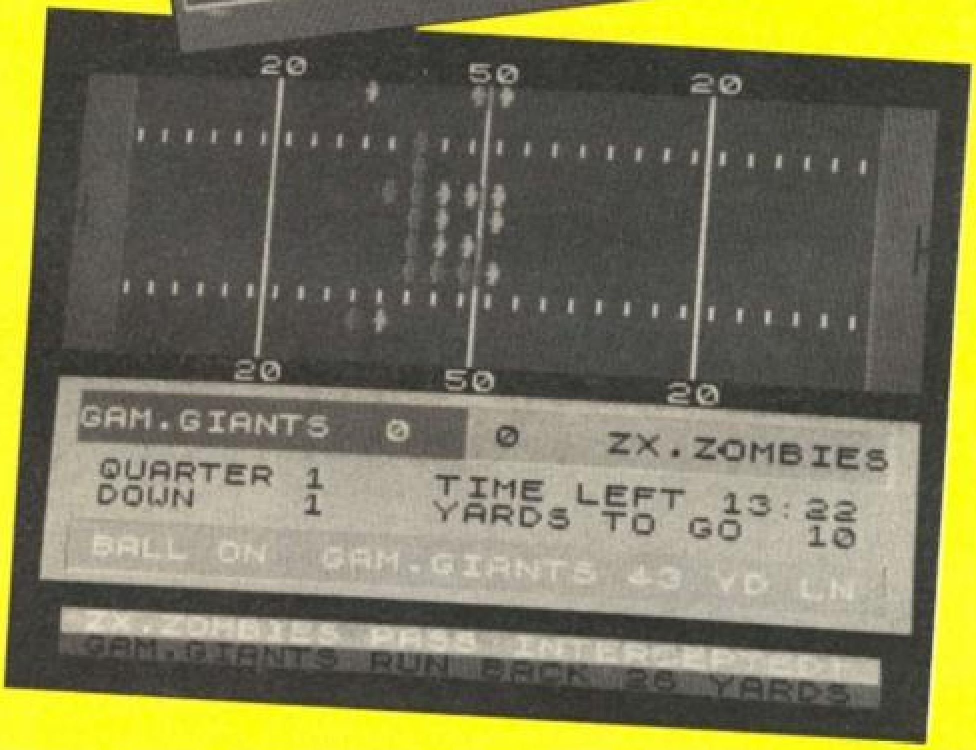
Listing 4

```
 1 REM **BASIC DRIVER PROGRAM**
 2 REM
 3 BORDER 0: CLEAR 64743: LOAD ""CODE 65000
 4 REM ***SET UP RANDOM NUMBER TABLE**
 5 REM
 6 FOR i=64744 TO 64999: POKE i,INT (255*RND): NEXT i
 7 LET n=0: REM **INITIALISE COUNTER
 8 REM **MAIN LOOP**
 9 REM
10 LET n=n+1: POKE 65057,n: REM **UPDATE PICTURE COUNTER
15 INK 0: CLS : RANDOMIZE USR 65059: REM **BACKGROUND PAPER
20 PRINT AT 1,17;"PICTURE NUMBER";AT 3,24;n
30 INK 0: RANDOMIZE USR 65140: REM **MOUNTAINS
35 INK 7: RANDOMIZE USR 65300: REM **LAKE
40 INK 4: RANDOMIZE USR 65346: REM **REEDS
45 INK 0: RANDOMIZE USR 65407: REM **GROUND SPECKLE
50 INK 4: RANDOMIZE USR 65442: REM **GRASS
60 PRINT #1;AT 1,9; FLASH 1;"NEXT PICTURE?": PAUSE 0
70 GO TO 10
```

SPORTS SIMULATION OFFER







Three games for the price of one from Argus Press Software

Sports simulation fans can pick up three games for the special prive of £9.95 in this ZX readers offer organised in conjuction with Argus Press Software.

The most recent release is **Peter Shilton's Handball Maradona** a football simulation with a difference as it puts you in the boots of the goalkeeper with the challenge of keeping a clean sheet in practice games and league matches at a variety of skill levels.

Nick Faldo's Open gives you the run of 900 screens and 18 holes as you play round a faithful simulation of the course at Royal St George's. The package comes complete with a fully illustrated guide to the course.

American Football is an animated grid-iron simulation for one or two players featuring all the elements of the real game — passing offense, running, defence and kicking to create a blend of on field action and strategy.

Now these games are available by mail order for just £9.95 including postage and packing. Do not send your cheques or postal orders to the ZX address but to the address on the coupon on this page. Make cheques and postal orders payable to **Argus Press Software.**

But if sports simulations don't appeal to you don't despair this is just the first in a series of special offers featuring Argus Press Software games and forthcoming offers will feature war, space and arcade games.

Argus Press Software Sports Offer

Please send me Handball Maradona, Nick Faldo's Golf and American Football.

Name:...

Address:...

....

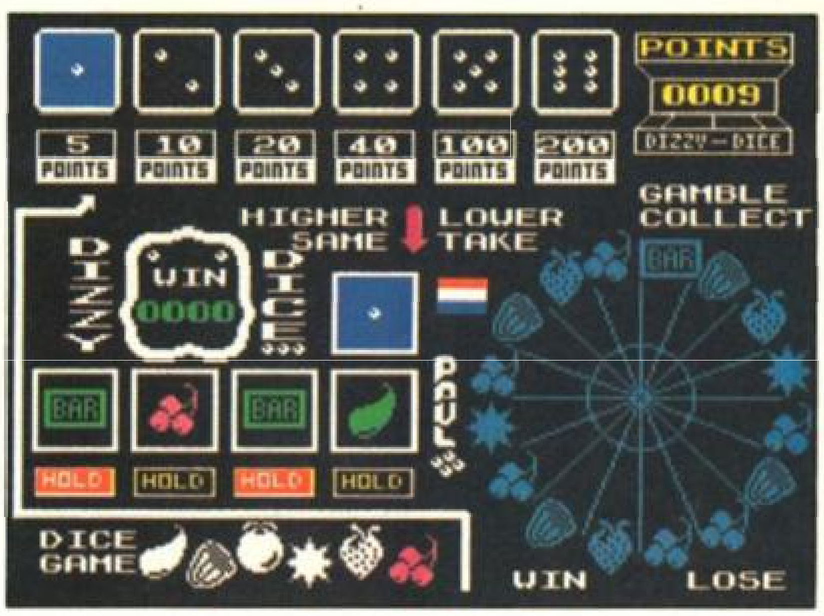
I enclose a cheque/postal order for £9.95 including p&p payable to **Argus Press Software.**

Send your remittance to: **Argus Mail Order (ZX), Units 1&2, Conlon Developments, Watery Lane, Darwen, Lancashire BB3 2ET.**

GAMES

DIZZY DICE

Players
£1.99



Fruit machine simulations inevitably lack the excitement of the real thing. On the plus side you can't lose your shirt but on the other hand you're not going to walk away with your pockets bulging with loose change. Without the thrill of the gamble the simulation is ultimately — fruitless. That being said Dizzy Dice manages to be slightly better than most games of this kind so if you want a harmless imaginary flutter this might be worth a whirl especially at a budget price.

The presentation is bright and full of activity once the fruits are rolling. A touch of diversity is added to the usual game with the opportunity to gamble your winnings on a roulette wheel or on the throw of a dice (guessing whether it will be higher or lower than the previous throw).

Starting off with $10 you must turn it into $100 thus breaking the bank. You then move on to try breaking the bank at higher cash limits.

Dizzy Dice is about as good a game as you'll get in this unadventurous genre but purists will notice a glaring inaccuracy. When they are about to run out of coins, the machine will suddenly turn benevolent and give you four fruits in a row and a hefty payout, just to keep you in the game.



WIBSTARS

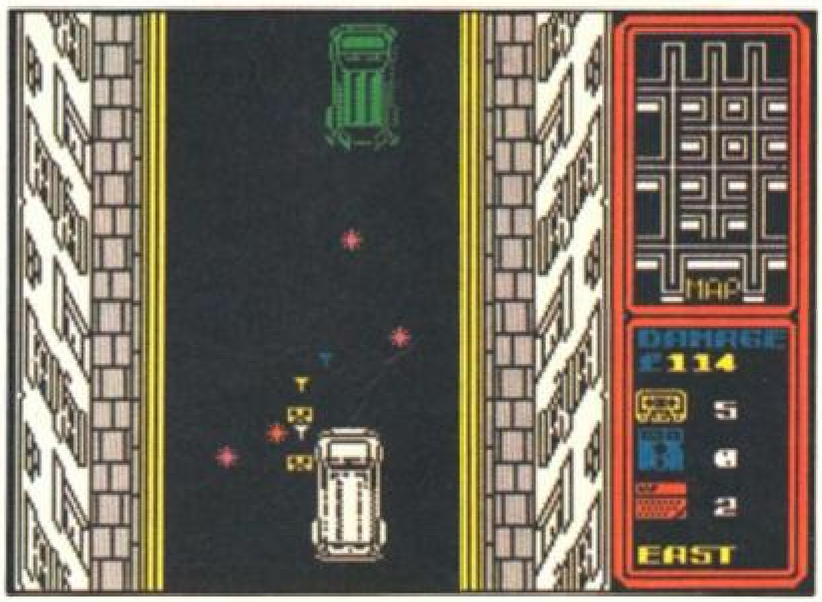
A'n'F
£8.95

Somebody somewhere thought it would be a great idea to base a game on the trials and tribulations of a software distribution company. With such an inspired concept as a starting point this game was always going to face an uphill struggle to succeed. Nonetheless great games can spring out of lacklustre notions — unfortunately Wibstars isn't one of them.

Wibstars is three mini games tacked together. Firstly there's a very rudimentary game in which you must get your van under the right chute to receive the goods due for despatch. Secondly there's a brief arcade section, where your van is displayed from a birdseye view going along the road. In front is a van from a rival firm which jettisons debris into your path. All you have to do is avoid the junk thrown at you. The climax of the game is a platform and conveyor belt scene — manoeuvre your goods up the screen, avoid the hazards and you can finally get your goods to market. As a screen from a platform game it works reasonably well but there's only one. Any platform game you could mention has far more screens of at least equal complexity.

Wibstars three games do not stand up either on their own or when thrown together in this order. A poor release which is below the standard of much budget software.





AGENT ORANGE

A'n'F
£8.95

This is the famous "life" program of multiplying cells transplanted and grafted onto an arcade game. The result is a farmers in space foray with liberal doses of sowing, reaping and zapping.

The aim of the game is to establish plantations on alien planets. There are eight to conquer and the final planet has vast supplies of weedkiller with which you can rid the galaxy of weeds. Apart from agriculture there's an element of trading — on returning to your home planet you can cash in your crops and buy a bigger ship. The main action however takes place on the uncultivated planets where you pilot your tiny ship out of the mother craft and set about seeding the surface. There are alien fighters to avoid and destroy and indigenous weeds that will choke your harvest. counteract these threats, harvest your crops and fly back to the mothercraft. That's about the extent of the gameplay — there's a little bit of everything but no highly addictive feature.

Agent Orange is an interesting attempt to try something a bit different but the idea of combining "life" with arcade action sounds a lot better than it plays in practice.





FEUD
Mastertronic
£2.99

Released on yet another of Mastertronic's labels (Bulldog) Feud is a promising looking game that doesn't quite come up with the goods.

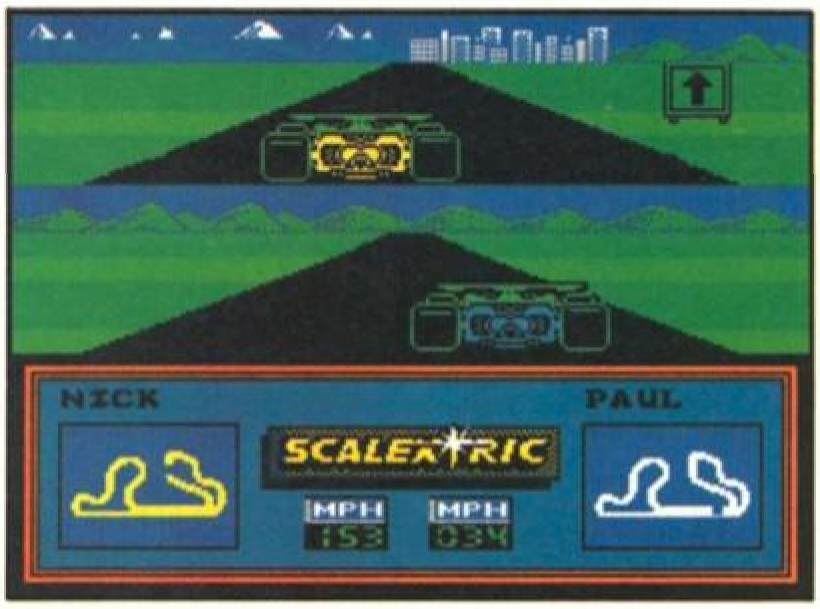
The plot has possibilities — you play a wizard named Learic who is involved in a feud with his brother Leanoric, who also happens to be a wizard. Being wizards their feuding involves much turning of each other into frogs, blasting with fireballs and the like. But before the spell throwing you both have to collect the necessary herbs and ingredients to prepare the spells.

Graphically the game is very slick. Learic is quite a large figure, very well animated as he wanders around in his monk's habit, and on the whole the graphics compare well with quite a few full-price titles. The trouble is that the business of finding all those herbs and things goes on a bit too long, so that instead of getting down to the serious business of transmogrifying each other, you spend most of the time wandering around bits of forest and along dead-end pathways looking for flashing flowers. Finding the herbs ought to be just the build-up to the big fight, but as it is it takes so long that the action gets slowed down and drawn out, making the game a lot less addictive than it could be.

Still, it is very nicely presented, and while it's not a budget classic it is a fairly respectable effort.







SCALEXTRIC
Virgin/Leisure Genius
£9.95

Scalextric arrives on the Spectrum as a sophisticated construction set with almost limitless possibilities for designing your own tracks as well as 17 pre-set simulated Formula 1 circuits.

Constructing a circuit is simplicity itself with an icon driven system and there is the option to save the tracks you've built. Racing takes place on a split screen display where you have the option to race in two player mode or against a computer controlled car. There's also a plan view of the whole circuit which plots your position during a lap — a useful aid when trying to anticipate sharp bends.

This computer simulation offers plenty of variety as far as tracks are concerned but because every race is just a two car race the racing action can become stale very quickly. It may have been an idea to change the Scalextric formula and include a larger field of cars, weaving your way through back markers can be just as satisfying as winning.

One area of racing that has at last got some attention is crashing. In most simulations whatever happens in one player mode it's always your fault and you are eliminated, whereas in Scalextric the rule is that the car approaching from behind is always the loser, but if the difference in speed between the two colliding cars is less than 20 miles an hour there's no crash — the front car is just shunted along the track.

A good value package only lacking the competitive edge that comes from having a grid full of cars racing round the track.



BUTCH HARD GUY
Advance Software
£7.95

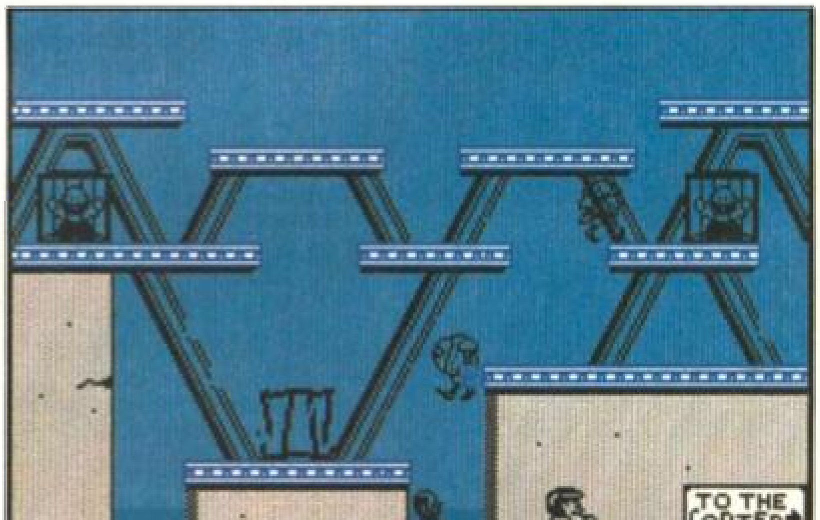
It's wild: It's wacky! It's, no it isn't, yes it is. . . it's a platform game.

Butch Hard Guy is, believe it or not, a spoof of the recent spate of Rambo, Commando, Cobra types of game — in other words it's just another combat game with a few funny names thrown in. Spoofs are meant to be funny, but in this case the hilarity begins and ends with the names of Butch himself and his opponent, the evil Dr. Tie Fu (that'll have 'em rolling in the aisles). Once you've recovered from the rib-splitting uproariousness of all that you've got nothing left to do but play the game.

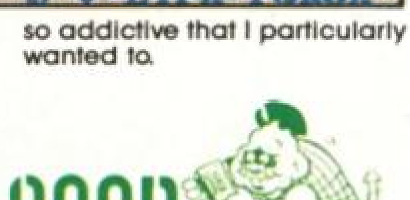
Butch is an old war veteran whose task is to rescue a number of other vets who have been captured by Tie Fu and locked up in his castle in the south Pacific. The cages that they've been locked in are situated n platforms on each of the games 20 screens, and, as Butch, you have to leap from platform to platform, kicking the cage doors open and enabling the prisoners to escape to Butch's helicopter (see, I told you it was *a platform game).*

Once all the prisoners have escaped from a screen you can then start on the next one, but there is of course a slight snag. Each screen is patrolled by Dr. Fu's robot guards, which are coated in a poison which kills on contact. Not only can these robots kill Butch, they can also recapture the prisoners which means that he'll have to go back and rescue them again unless he's very fast. You can defend yourself from the robots by either punching them, which just makes them sit still for a second or two, or kicking them, which can blow them up (but you'll have to get out of the way or be caught in the blast).

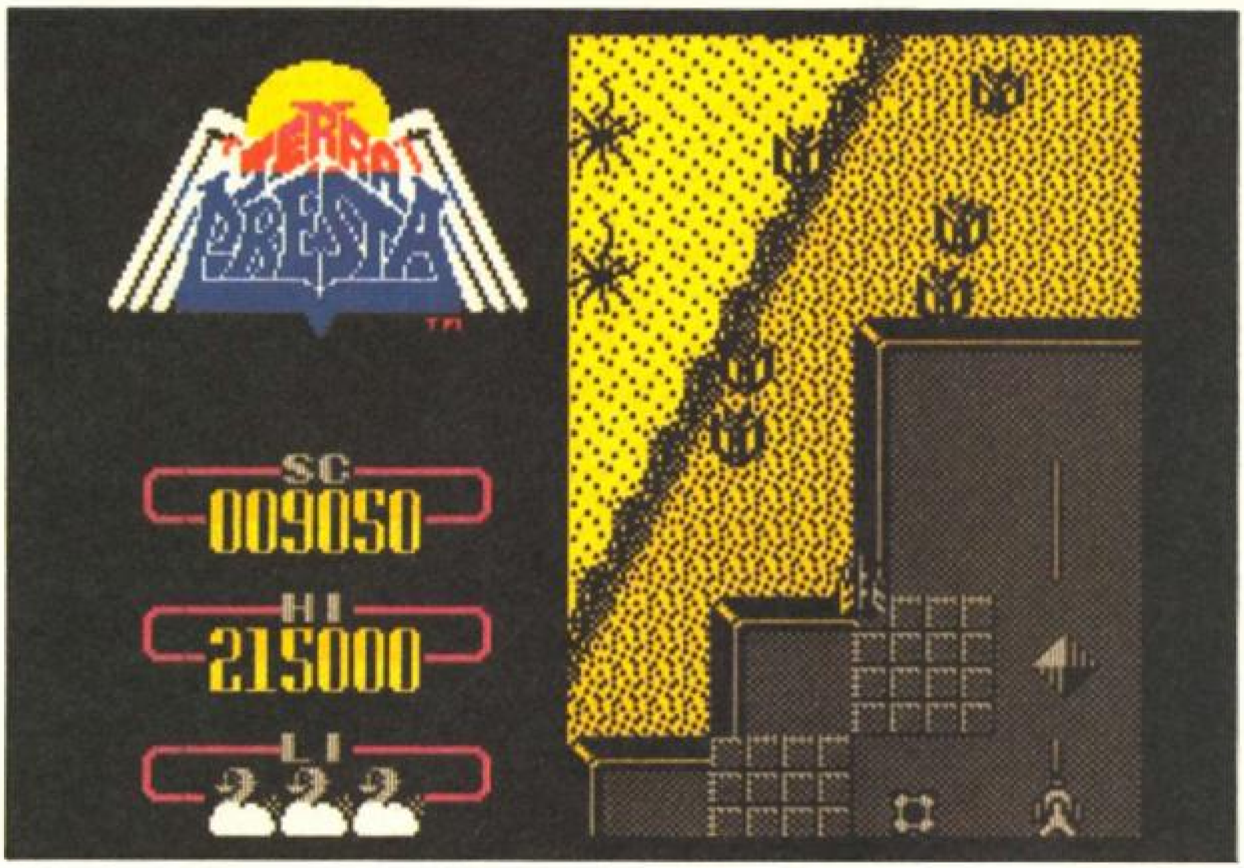
The movement of the robots isn't predictable, and I found that you could attempt to hit one of them and still end up



dying, which made the game a little irritating at times. The twenty screens are arranged in a fixed order and you'd have to be amazingly quick-fingered to get to the end (and the game won't accept Sinclair interfaces, so if you've got one, or a +2 then you'll have to use the keyboard), but the game wasn't so addictive that I particularly wanted to.



TERRA CRESTA
Imagine
£7.95



Another arcade conversion from Imagine, this time of a space shoot 'em up that reminds me a bit of Gargoyle's Lightforce. You control a spaceship flying over the scrolling surface of a planet guarded (stop me if you've heard this one before) by wave upon wave of aliens who will swoop out of the sky and try to blow you all over the place. Plus, at the same time as dodging and firing back at the aliens you've also got to destroy the weapons silos on the ground which are adding to the fun by lobbing bombs into the sky.

Needless to say, all this sky-borne action keeps you very busy, but as you pick off the silos you are rewarded with additional weaponry that helps you blast your way through to the final stages of the game where you'll probably be trashed by a huge robot-ship.

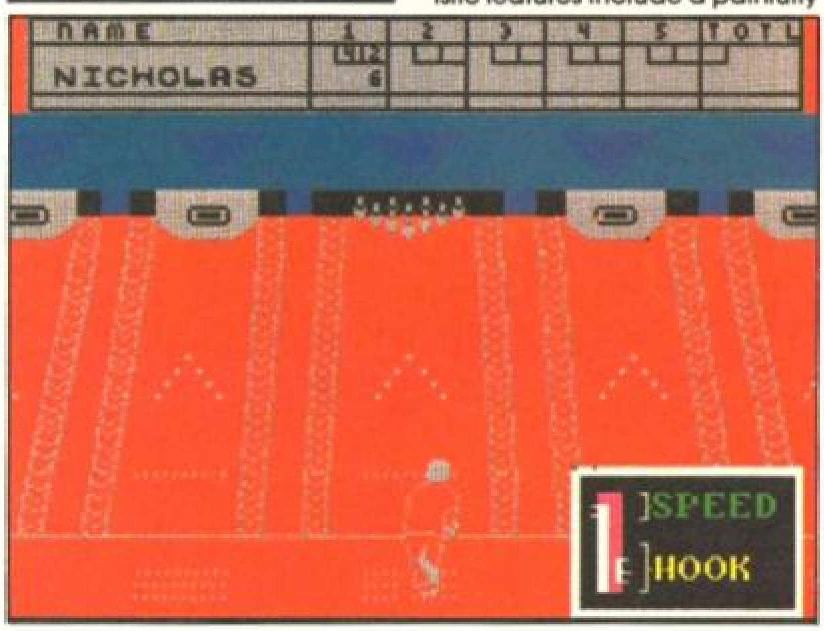
It's all fairly standard shoot 'em up action, professionally presented but not exactly oozing with originality. For some reason (probably to keep the conversion as faithful as possible to the arcade original) the playing area occupies only the right-hand side of the screen, while the rest is occupied by the Terra Cresta logo and score tables. As a result the graphics in the small playing area, though reasonably well drawn, are pretty small and require close attention to keep an eye on what's happening. I know it might mean a less faithful conversion, but it might have made a slightly better game if the programmers had used the whole of the screen for playing it — as it stands it looks a bit like you're only getting half a game since one half of the screen is virtually useless.

There are no major faults in Terra Cresta but without a bit more originality it's not likely to take over as anyone's favourite shoot 'em up.



GAMES

10th FRAME
US Gold
£9.95



When it comes to sports simulations, authenticity can be a drawback. This ten pin bowling game replicates the tedious features of the game while failing to exploit the excitement of the actual game. Tedious realistic features include a painfully long wait while the machine at the end of the lane sweeps away knocked over pins. Authenticity could certainly have been sacrificed to avoid the pauses especially as even during the action this game is still in the slow lane.

Controls are keyboard only and there is no computer opponent so you've either got to be content on improving your own score over the three difficulty levels or strongarm some friends into joining in (up to eight players).

The skill factor comes in during the aiming and throwing of the ball, direction is selected with a cursor and speed and accuracy by testing your reactions on a moving bar diagram at the bottom left of the screen. This indicator also determines the amount of "hook" you put on the ball. A bad error of judgement means that the ball will generally spin into the gutter about a third of the way down the alley.

Mastering the refinements of the game takes some time but once you've accomplished that it becomes very much like top level darts, it's more of a surprise when you miss than when you hit the target. Even when you get a strike it's very low key — no flashing lights or whistles.

The animation, such as it is, is good and the graphics are fair and although it will suit those who are ten pin bowling fanatics the impulse buyer may find it repetitious and overpriced.



FIRST STEPS INTO MACHINE CODE

BY ALAN DAVIS

Last month I described what I hoped was a painless method of making the often traumatic transition from BASIC to machine code programming. If you've had a go at it, then I hope it proved successful for you — but I must confess that I'd feel a little guilty if we just left it at that. Those who are left scratching their heads and wondering "How do I get from here to 3D sprite graphics?" might, with some justification, feel somewhat dismayed! So here goes with a second instalment. The 3D sprite graphics are still a bit far off, but I hope we can get a few steps nearer without offending the purists too badly.

Screen routine

What I want to do in this article is to build upon the ground that we cleared in the first one and introduce a few new concepts (but not too many). In so doing, we'll actually be able to write a rather useful little screen manipulating routine. The programming won't be elegant, it won't be economical, and it'll give Toni Baker hysterics — but it'll be simple (very), and it will work pretty well. Just to whet your appetite, take a look at Listing 1 — or, better still, drag the old Spectrum out of its corner and spend a couple of minutes typing it in. When you RUN it, the program will draw a pattern on the screen and ask you to set the PAPER and INK values. If you respond in each case with a number between 0 and 7, the screen will be coloured in from top to bottom in the new PAPER and INK colours, leaving the original pattern completely intact. You've probably seen this sort of thing before — it's often called a "wash" routine, and it would be a useful addition to any graphics designing program since the Spectrum doesn't have an inbuilt BASIC command to do it. The trouble is that, as you may have noticed, it's very s-l-o-w!!

We don't have to look far to find the reason for this snail's pace. It lies in lines 80, 90 and 100, where the program loops all the way through the screen attributes area of memory, poking into each address the appropriate value for the current choice of INK and PAPER. Since there are 768 character squares to be coloured, it's not surprising that this takes a bit of time. Wouldn't it be nice if we could replace lines 80 to 100 with a

Listing 1

```
  1 REM *** BASIC "WASH" ROUTINE ***
  2 REM
  8 REM *** DRAW PATTERN ***
  9 REM
 10 CLS : FOR i=1 TO 96
 20 PLOT 127,87: DRAW 80*SIN (PI*i/48),80*COS (PI*i/48)
 30 NEXT i
 37 REM
 38 REM *** PICK UP INK AND PAPER COLOURS ***
 39 REM
 40 PRINT #1;AT 0,0;"Paper colour? (0-7)"
 50 GO SUB 200: LET paper=VAL i$
 60 PRINT #1;AT 0,0;"Ink colour? (0-7)"
 70 GO SUB 200: LET ink=VAL i$: IF ink=paper THEN  GO TO 70
 77 REM
 78 REM *** NOW "WASH" THE SCREEN IN THE CHOSEN COLOURS ***
 79 REM
 80 FOR i=0 TO 767
 90 POKE 22528+i,ink+8*paper
100 NEXT i
110 BORDER paper
120 GO TO 40
197 REM
198 REM *** PICK UP A KEYPRESS ***
199 REM
200 PAUSE 0: LET i$=INKEY$: IF i$<"0" OR i$>"7" THEN  GO TO 200
210 PRINT #1;AT 0,0,,
220 RETURN
```

Listing 2

Use of the HL register pair to "point" to an address.

```
65368   10              ORG 65368
65368   20              LD HL,22528
65371   30              LD A,(COLOUR)
65374   40              LD (HL),A
65375   50              RET
65376   60 COLOUR       DEFB 7
```

Listing 3

A simple "counting" loop

```
65368   10              ORG 65368
65368   20              LD A,0
65370   30              LD (COUNT),A
65373   40 LOOP         LD A,(COUNT)
65376   50              CP 255
65378   60              RET Z
65379   70              INC A
65380   80              LD (COUNT),A
65383   90              JP LOOP
65386  100 COUNT        DEFB 0
```

FIRST STEPS INTO

machine code routine to do the same job instantly? After all, the Spectrum's screen attributes area is just a section of memory like any other (it occupies the 768 bytes from address 22528 onwards), and as we know, shoving numbers around like this is something that the Z80 is very good at. So this is what we're going to attempt.

Registers

In the previous article we dealt with the Z80's registers only one at a time, but a single register can hold only numbers up to 255, and to write our "wash" routine we'll need to be able to handle numbers larger than this. Fortunately there's a simple solution. We can store larger numbers by using TWO registers, and the Z80 has a large set of instructions for handling pairs of registers for just this purpose. Two registers which are paired together are the H and L registers (other common pairings are D/E and B/C), and an instruction for loading the HL pair with a number takes the form:

LD HL,515

What's actually happening here? Well, the number 515 is in fact being split into two separate bytes. One of these — the "H"igh byte — stores multiples of 256, and the other — the "L"ow byte — stores whatever remain. So in this case the H register stores the number 2 (2x256=512) and the L register stores 3 (i.e. 515—2x256). It's easy to discover the biggest number that can be stored in this way: if both registers contain 255, then we have 255+255x256=65535. However, we don't really need to cope with this in detail at present; it's enough just to appreciate that any number of 65535 can be loaded into the HL register pair.

We're now in a position to take a look at Listing 2, which is an assembly program for changing the attributes of the first character square on the screen (top left hand corner). We'll be storing the machine code in the user defined graphics area, and the ORG directive at line 10 tells the assembler to do this. Line 20 loads the number 22528 (which is the address for the start of the attributes area, remember) into the HL register pair, and line 30 loads the A register with the contents of the byte labelled COLOUR (see line 60, where the defined value of 7 corresponds to white INK and black PAPER). This much is fairly obvious — but now comes the clever bit, and a new Z80 instruction: LD (HL),A. This means "put the number held in the A register into the byte of memory whose address is held in the HL register pair." In our case, this results in the number 7 being stored in address 22528. Then, finally, line 50 returns us to BASIC.

Now you might be thinking "but why don't we just use the instruction LD (22528),A instead of all this messing about with HL?". But don't worry about that for now — all will be revealed in due course. Meanwhile, you might like to try out this little routine for yourself. To do so, assemble the code and return to BASIC. NEW the Spectrum (which will still leave the machine code intact in the UDG area) and enter the command PRINT AT 0,0;"A" to get the letter "A" in the top left character square. Now type **RANDOMIZE USR 65368,** and abracadabra — the "A" will still be there, but in white INK on black PAPER. Try poking a different value (8xPAPER+INK) into 65376 (COLOUR), and do RANDOMIZE USR 65368 again. It really does work, doesn't it? So all we need now is to repeat the process 768 times to change the colours for the whole of the screen; in other words, we need a program loop.

Loops

Forget our present problem for the moment, and let's consider a simpler one. How can we make the Z80 simply count a specific number of cycles (say 256) and then "return to BASIC"? In other words, can we write the equivalent of something like:

FOR I=0 TO 255: NEXT I

Well yes we can. There are several ways of doing it, in fact — and Listing 3 is one of them. There are better methods than this one, but this method perhaps sticks closer to the concepts we already know; actually, a better equivalent as a BASIC subroutine would be:

```
10 LET COUNT=0
20 IF COUNT—25=0 THEN
RETURN
30 LET COUNT=COUNT+1
40 GOTO 20
```

What I've done in Listing 3 is to set up a single byte (COUNT) which will serve the same purpose as the counting variable in the little BASIC loop above. The program proper starts at line 20, by clearing out the contents of the A register, and line 30 then dumps the result (zero) into the address labelled COUNT. This is just to make sure that the counting of the loops does actually start at zero. At line 40 we enter the loop itself — and I've used the label LOOP to mark this point in the program; we'll see why in a moment. Here we put the current contents of the COUNT byte into the A register (zero on this first pass, of course), and line 50 then confronts us with another new instruction: CP 255. Think of this as meaning "ComPare the number in the A register with 255". Now obviously the Z80 can't actually "compare" two numbers in any intelligent sense, but it can do a simple subtraction of one from the other — and if the result of the subtraction is zero, then clearly the numbers are the same. So the outcome of the CP 255 instruction will be "remembered" by the Z80 as either ZERO or NOT ZERO. (Incidentally, this result is stored in a single bit of the F register called the "zero flag", but you don't really need to worry about that here). The contents of the A register are unchanged by all this.

I think you'll see that here we have the possibility of conditional actions. We can make the program perform one action in the event of a ZERO result, and another if we get NOT ZERO. Which brings us to line 60 — and you can probably guess what "RET Z" means: "RETurn if the result is Zero". In other words, if the number held in the byte COUNT turns out to be equal to 255, the program will return to BASIC. On this first pass, of course, COUNT contains zero — it isn't equal to 255. So the program moves on to the next instruction, line 70. Here the A register is increased by 1 (we dealt with INC A last month), and then line 80 causes this new number to be deposited in COUNT. After the first pass, then, COUNT will no longer contain zero, but 1. Finally, at line 90, we encounter a "jump" instruction: JP LOOP. This tells the Z80 to go back to the instruction labelled "LOOP" and carry on from there. I think you can follow the rest yourself. The program will cycle round and round, increasing the contents of COUNT by one each time, until eventually it contains 255. When this happens, line 60 will cause a return to BASIC.

This is the most complicated bit of programming we've attempted so far, and it's worth making sure that you understand just what's going on before proceeding further. If you get lost, refer back to the little 4-line

MACHINE CODE

BASIC loop counter for help. The parallel between that and the assembler program is very close.

Happy? Good — because we're now very nearly home and dry. If we combine Listing 2 and Listing 3, with just a very slight (but crucial) addition, we have a routine (Listing 4) which will perform a "wash" over the first 256 character squares of the screen. Lines 20 to 40 set up the initial conditions (put first address of attributes area into HL, and zero the counter) and then the loop is entered. The A register picks up the "colour" (line 50) and transfers it to the address which HL points to (line 60) Then we check the counter (lines 70 to 90) increase it by one (line 100) and store the new count (line 110). Now comes the crucial addition I mentioned. It's a new instruction, but you'll know exactly what it means: INC HL (line 120). This increases the number held in HL by 1 — so that HL now "points" to the next address in the attributes file. Line 130 causes a jump back to the instruction labelled LOOP, and the whole process is repeated over and over again until 256 cycles have been completed, causing a return to BASIC at line 90.

Can you see now why we used the apparently roundabout routine in Listing 2? By storing the current attribute address in HL, we can simply "point" the routine to the next address by using the INC HL instruction repeatedly as we go round the loop.

Listing 4 will indeed work — but of course it will only wash the top third of the screen. What about the other two thirds? No problem. All we need to do is to use the routine three times in succession, with HL initially containing 22528 for the first run, 22784 (22528+256) for the second, and 23040 (22784+256) for the third and final run. But we don't need to type in the whole thing three times, because the Z80 allows us to use the equivalent of a BASIC subroutine using the CALL instruction. Listing 5 — which is, at last, our ultimate goal — shows how it can be done fairly painlessly. What I've done here is to take the whole of Listing 4 (with the initialisation of HL removed), and attach the label "WASH" to it — as you can see at line 90 of Listing 5. This now forms a subroutine which can be called using CALL WASH.

The main assembler program then becomes the sequence of instructions from lines 20 to 80, in which HL is given the appropriate initial value, and the subroutine WASH called three times before the return to BASIC is made at line 80. Just one further point may puzzle you here: why is it that previously the RET Z instruction has caused a return to BASIC, whereas now it no longer does? The short answer — which is all we have space for — is that in this case it was preceded by a CALL instruction, and the Z80 "remembers" where to carry on from after it returns from a subroutine. Incidentally, just as every BASIC subroutine must end with a RETURN sooner or later, so every machine code subroutine must somewhere terminate with a RET.

Run

Now let's make it all work. Assemble Listing 5, and if you like, save the machine code bytes thus produced (SAVE "wash" CODE 65368,44). NEW the Spectrum (which leaves the machine code still intact in the UDG area), type in Listing 1 again — and just for fun, run it once more to remind yourself what life used to be like before you were a machine code wizard. Tedious, eh? Now BREAK, delete lines 80, 90, and 100, and add the following BASIC lines:

80 POKE 65410,ink+8*paper
90 RANDOMIZE USR 65368

Now RUN the program. Wait for the pattern to be drawn; choose your paper colour; hold your breath ... then choose the ink ...
Magic.

Listing 4

"Washing" the top third of the screen.

```
65368  10          ORG 65368
65368  20          LD HL,22528
65371  30          LD A,0
65373  40          LD (COUNT),A
65376  50 LOOP     LD A,(COLOUR)
65379  60          LD (HL),A
65380  70          LD A,(COUNT)
65383  80          CP 255
65385  90          RET Z
65386 100          INC A
65387 110          LD (COUNT),A
65390 120          INC HL
65391 130          JP LOOP
65394 140 COLOUR   DEFB 7
65395 150 COUNT    DEFB 0
```

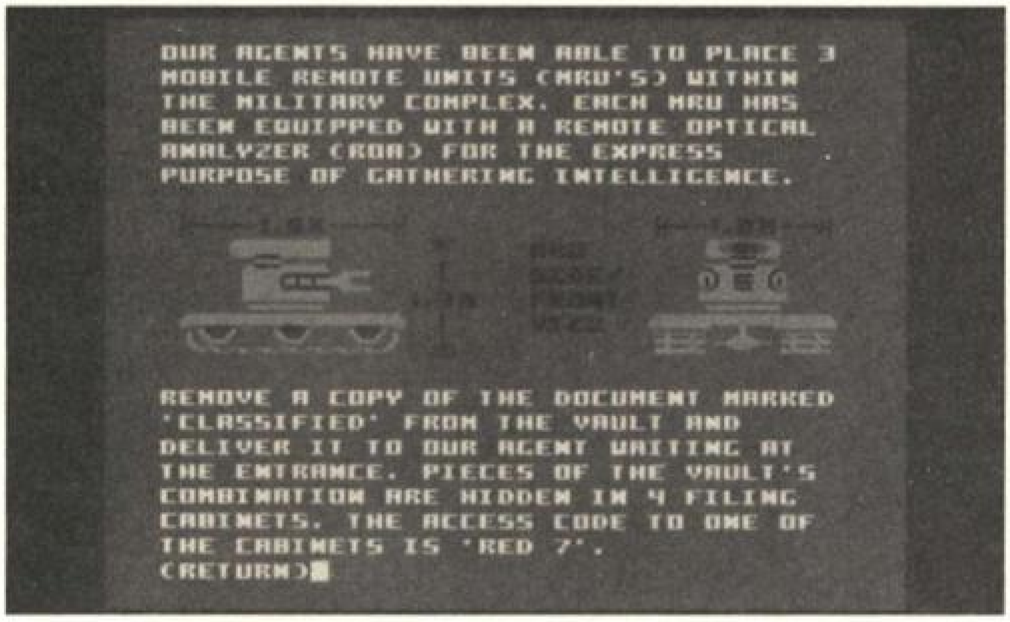
Listing 5

Complete screen "wash".

```
65368  10          ORG 65368
65368  20          LD HL,22528
65371  30          CALL WASH
65374  40          LD HL,22784
65377  50          CALL WASH
65380  60          LD HL,23040
65383  70          CALL WASH
65386  80          RET
65387  90 WASH     LD A,0
65389 100          LD (COUNT),A
65392 110 LOOP     LD A,(COLOUR)
65395 120          LD (HL),A
65396 130          LD A,(COUNT)
65399 140          CP 255
65401 150          RET Z
65402 160          INC A
65403 170          LD (COUNT),A
65406 180          INC HL
65407 190          JP LOOP
65410 200 COLOUR   DEFB 7
65411 210 COUNT    DEFB 0
```

Activision's latest casts the hacker as hero.

HACKER 2



Hacker 2: The Doomsday Papers
Activision
£9.99

A cassette labelled Actisource, an incomprehensible technical manual and no instructions would not seem the obvious beginnings for a great game. But to my great surprise, these ingredients make Hacker 2 one of the most sophisticated releases I have played for ages. It's American, of course.

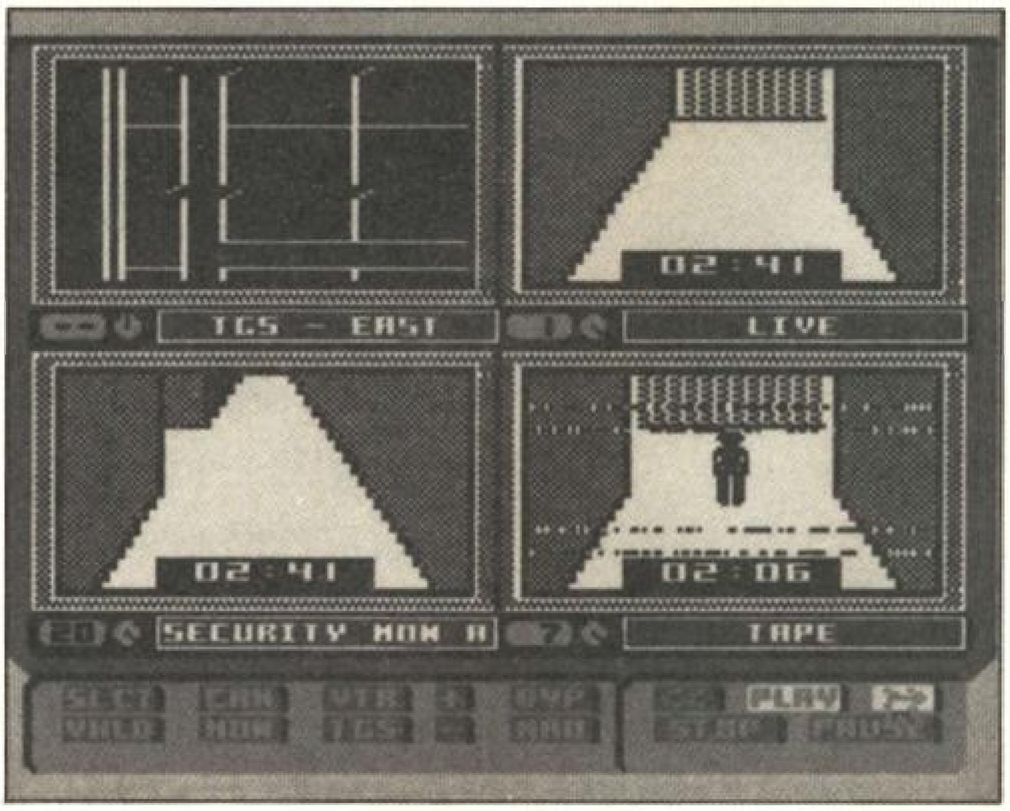
In this game's prequel, you hacked your way through numerous obstacles to prove yourself one of the world's finest hackers, but your endeavours did not go unnoticed. Loading the Actisource tape, you log on to what seems a normal enough Micronet style service — until the menu is interrupted by a message from the CIA! As the leading expert on breaking security systems, they have a mission for you. The Russians have developed a simple but devastating plan to topple the US government (trust those scheming commies). The details are enclosed in a document — "The Doomsday Paper" (sort of thing the Ruskies would call such devious plans). Your task is to retrieve these papers from the military complex in Siberia (where else? As you may have guessed, I do not entirely approve of the typically paranoid American scenario). How can you do this, if you're sitting in front of a screen back home? Via the miracles of modern technology, that's how. Hidden within the base are three Mobile Remote Units (MRUs, droids at your disposal), and the Multi-Function-Switching Matrix (MFSM), which controls the MRUs. The MFSM is linked by satellite to CIA's central computer, which is connected by modem to your computer. On your screen you see a mock-up of the MFSM.

Security at the base is not so slack that MRUs can wander around undetected. Guards patrol the corridors, and there are thirty-eight security cameras scattered around. Two electronic monitors flick through these successively, checking that no unidentified objects are in the picture. With the MFSM, you can view what any of the 38 cameras see, what either monitor is registering, or a plan of the immediate area surrounding the MRU. The clever bit is, the MFSM also includes 38 video channels. If you synchronise correctly, you can bypass a camera with a recording of what it should be seeing, and sneak your MRU in unnoticed. You have to check several things at once, so the MFSM has four screens.

Sounds complicated? I've simplified it considerably! Hacker 2 is an extremely complex piece of software; it's difficult too, and challenging, though not especially intellectual. Quick thinking and intelligent mapping are required. What makes it so unusual is the incredible attention to detail. For example, you have to set the vertical hold on each of the TV screens. The video simulation is so authentic you even get noise bars when you fast forward. And the CIA manual is convincingly confusing (not that I've ever seen a CIA manual). Real efforts have been made with this game to create realism: something very rare with British software.

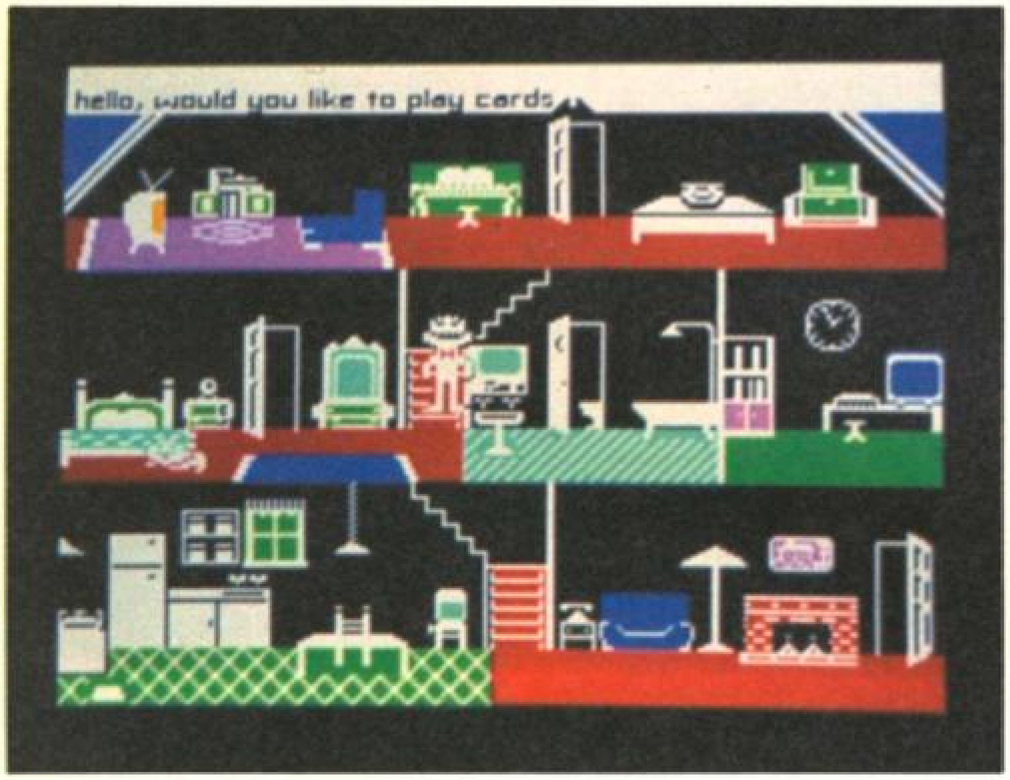
The few animated graphics are adequate rather than stunning (slightly chunky); otherwise Hacker 2 is perfectly executed. Patience is required, but it does pay off. Lastability is questionable, but I reckon this will provide value for money for most people. Activision's Spectrum conversions are usually lacklustre affairs (remember the lamentable Ghostbusters?), so congratulations are in order; at last they have created a winner!





Little Computer People

Now you can invite Little Computer People to take up residence in your 128.



Little Computer People
Activision
£9.95
(128 only)

Originally sighted inside the Commodore 64 it now appears that the Little Computer People have started to crop up in other places as well, namely 128K versions of the Spectrum (apparently there's not enough room inside the 48K models — they may be Little Computer People, but they're not *that* little).

But what are these little beings (LCPs as they are sometimes called?). The first LCPs were observed by American programmer, David Crane, who had suspected the presence of tiny human beings within his computer for some time when he created what he called a 'house on a cassette'. Loading this cassette into the machine created a small home inside it, empty and just waiting for someone to come along and live inside.

When you load your own 'house' into your 128 or Plus Two you are first of all presented with a notepad into which you enter your name, the date, and time of day. This, and other information about your LCP is continually updated and carried over from session to session.

Once we had gotten our own Plus Two loaded up, we all found ourselves sitting in front of the Spectrum, staring at the empty house on the screen. After a couple of minutes a small head stuck its way around the front door, then an LCP entered to give the place the once over.

This was Drew, a cool looking LCP sporting sunglasses and bow tie, who proceeded to take up residence (accompanied by his Little Computer Dog). Once your LCP is in, you've got a choice: you can sit back and do nothing and just watch him (yes, *him*, I've yet to hear of any sightings of LCP-esses which, by implication, gives rise to all sorts of interesting questions) wander around, or you can send messages or perform various actions via the keyboard.

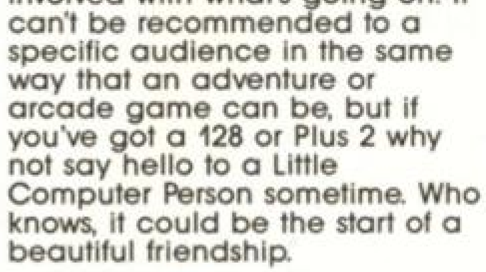
Left to himself, Drew seemed perfectly happy to go about the house cooking, listening to his stereo, tinkering with his computer or even doing aerobics. You can sit and watch for ages and your LCP will get on with things on his own. Watching this is a bit like playing with worry beads — ultimately pointless, but quite calming somehow.

However, like all intelligent beings, your LCP has certain needs which have to be met — food and drink have to be supplied by you, via the keyboard, and when he gets a bit or lonely it's up to you to cheer him up as best you can (perhaps by playing cards or anagrams with him).

There are certain actions which you can always rely upon, such as giving him a new book or record to play, but there is enough scope within the program to allow you to experiment and discover how best to get on with your LCP just as you would do with any other person.

Remember though, that if you neglect your LCP he's likely to go green and take to his bead with a terminal cause of the sulks, so it's your responsibility to look after him.

Little Computer People is thoroughly ingenious and a genuine oddity. It's an entertainment of sorts, though it's not a game. In a way it can almost be described as the computer equivalent of background music, except that it offers you the opportunity to get involved with what's going on. It can't be recommended to a specific audience in the same way that an adventure or arcade game can be, but if you've got a 128 or Plus 2 why not say hello to a Little Computer Person sometime. Who knows, it could be the start of a beautiful friendship.



Amphibian daring in the dungeon as Hewson unveil a sprawling magical epic

Ranarama
Hewson
£7.95

anarama follows the exploits of a rather luckless sorcerer's apprentice whose experiments with the potions have transformed him into a frog. The one advantage of his new found form is that he avoids being destroyed by an invasion of evil warlocks. As Mervyn you must try to wreak revenge on the warlocks and steal their magic. One each of eight levels ther are 12 wizards to confront in combat. Once you've located a warlock and hopped in his path you are transferred to a sub game where battle commences. Adding sub games as an extra to the main action has become something of a Hewson trademark in such games as Paradroid, Quazatron and Firelord and here we are faced with an anagram game where you must rearrange the scrambled letters of "Ranarama" back to the original word. This is a reflex test that calls for a keen eye and good co-ordination with the joystick. On each successive level the time limit is lessened. The style of lettering can make it difficult to distinguish the letters at first but one quickly gets accustomed. If you are in any way dyslexic however this part of the game could present an insoluble puzzle.

If you succeed in sorting out the letters in time, the action reverts to the main screen where the warlock has disappeared to be replaced by hovering runes which you must collect quickly before they vanish. Amassing these runes is essential as they can be converted into magic powers. Inserted in the floors of chambers are hieroglyphs with a variety of functions — head for the "Glyph of Sorcery" and once activated you can scroll through the spells which are available to you with the runes you have. Some spells offer powers that won't come into effect until a later level. At you only have two lives it's best to concentrate on obtaining spells that will get you beyond your present level.

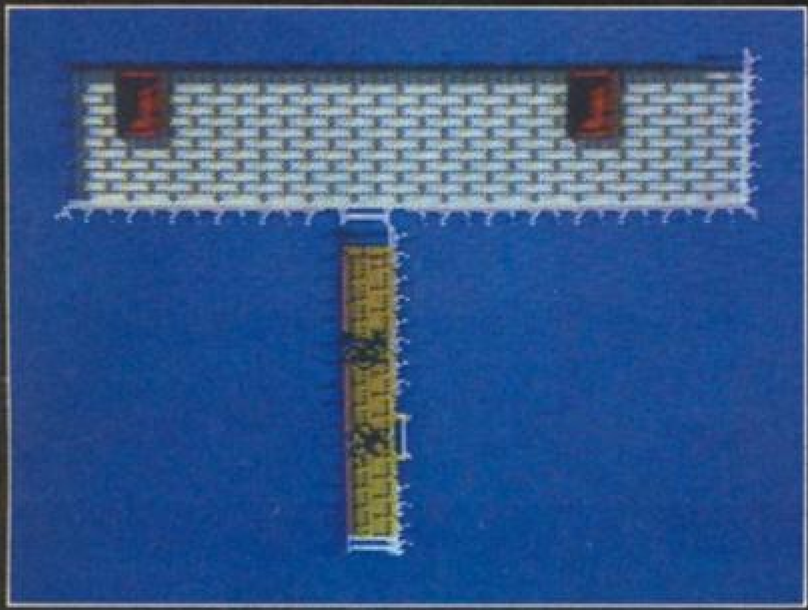
Seeing the light

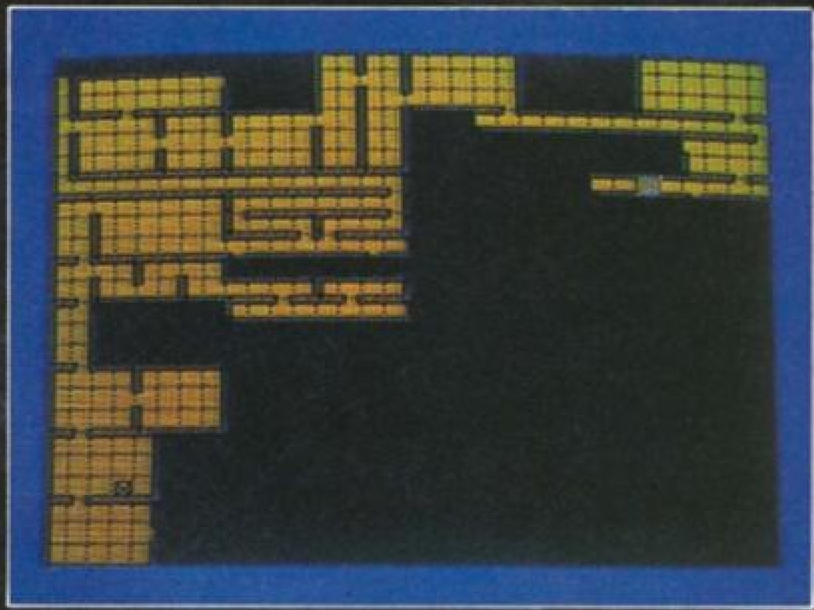
Each of the eight levels contains between 50 to 100 rooms and a very simple but effective system makes the usual meandering round a lot more enjoyable. At the start of each game you are placed randomly somewhere on a level — go through a door and the room you enter lights up. As you progress round, locate a "glyph of seeing" which will enable you to get a view of the entire level (but only those rooms you've been in will show up). On this map you may be able to locate hidden doors. In the dungeons themselves hidden doors can be found by taking advantage of a neat visual touch — figures peek their heads through the walls for a fraction of a second revealing where a door is placed.

As you would expect the dungeons are packed with various nasties like serpents, dwarf warriors and hideous insects which require different levels of firepower to be destroyed. These will all drain your energy but there is a floor glyph which can be activated to clear a room of creatures. This glyph is disposable so you can only use it once. Perhaps a greater problem, are the magical weapons like spinning swords and munching mouthes which make life especially difficult as they are impervious to attack. You can however destroy the generators that create them.

Even if you have managed to light up all the rooms on a level you may not have found all the wizards, defeating all twelve is essential to progress and as they pop up where they feel like it



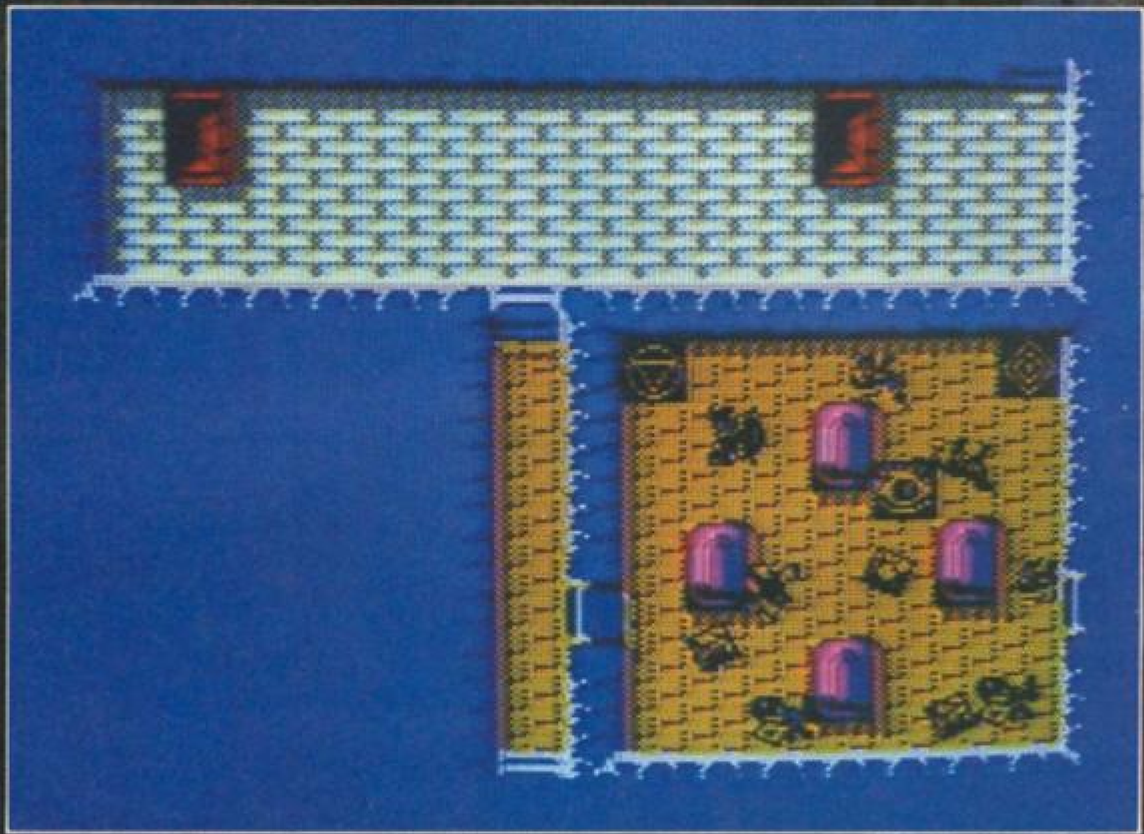




you will need to save some of your runes to purchase the spell that will reveal their location.

Ranarama is a highly accomplished example of games programming with just the right degree of zapping and strategy. There are obvious comparisons with Gauntlet and other arcade adventures but Ranarama has a character all its own as well as having all the necessities — it's fast and furious with a vast area and most importantly it's very playable and once you've got your bearings, very addictive indeed.

MONSTER HIT





LEADER BOARD

The most authentic golf simulation the Spectrum has ever seen.

Leaderboard
US Gold
£9.95



Sports simulations have been in the doldrums recently so the arrival of Leaderboard (a chart-topping Commodore game) on the Spectrum is cause for celebration especially as it's in a different league to any existing golf game.



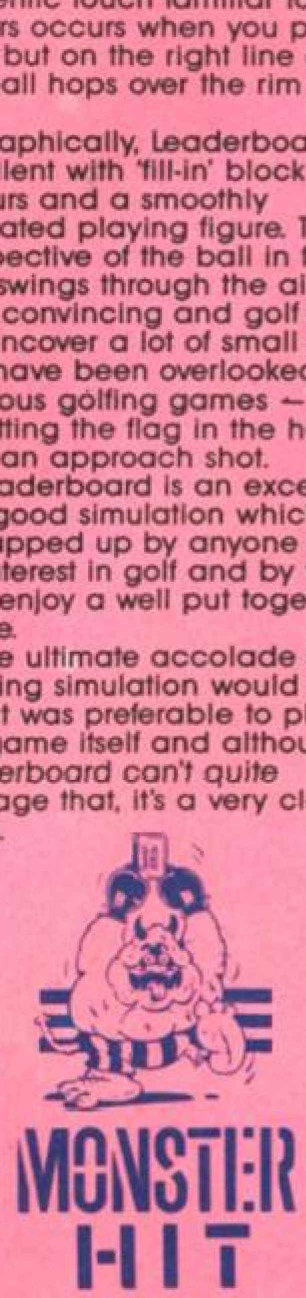
The list of options is undoubtedly impressive — there are four 18 hole courses to play on plus a practice driving range. Up to four players can take part in a round and there are three levels of difficulty, novice, amateur and professional (where subtle shifts in wind direction affect the flight of the ball). Each player has a choice of club (from an available 14) and can adjust the power, slice and hook in his shot by timing the drive against a moving bar display.

The joy of the game is the high responsiveness to subtle changes in your play. Select the wrong club or apply too much slice and you pay dearly as your score threatens to expand off the scoreboard. On a lot of holes, water hazards dominate and if this was real life you'd be spending a fortune in lost balls. Pinpoint accuracy is essential all the way down to the hole. You are kept up to date with the exact distance to the hole, even down to the last few inches so that even a tap in can be a problem especially as most greens have a vicious slope. An authentic touch familiar to poor putters occurs when you putt too hard but on the right line and the ball hops over the rim of the hole.

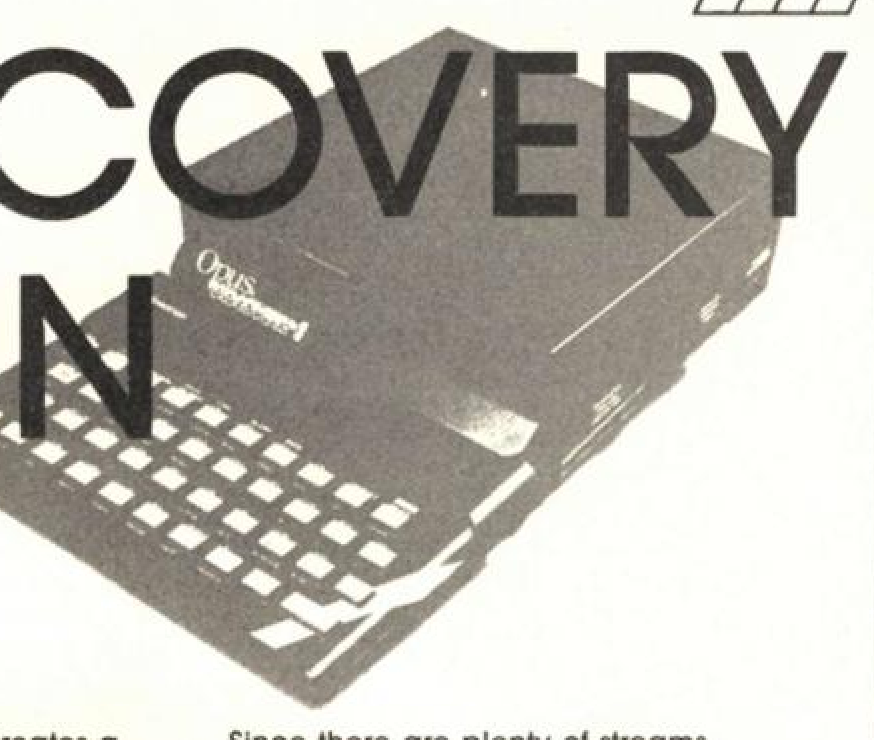
Graphically, Leaderboard is excellent with 'fill-in' block colours and a smoothly animated playing figure. The perspective of the ball in flight as it swings through the air is most convincing and golf buffs will uncover a lot of small details that have been overlooked by previous golfing games — such as hitting the flag in the hole from an approach shot.

Leaderboard is an exceptionally good simulation which will be lapped up by anyone with an interest in golf and by those who enjoy a well put together game.

The ultimate accolade for a sporting simulation would be that it was preferable to playing the game itself and although Leaderboard can't quite manage that, it's a very close thing.



THE DISCOVERY COLUMN



John Wase presents another selection of useful routines for Discovery owners.

Some months ago, a correspondent was asking about random access files and how to handle them. He thought that perhaps the best solution was to set up a relatively short file of key numbers to be used as an index to manipulate larger files. When I wrote back, I pointed out that this was a general disc filing problem rather than a Discovery problem, but that he was on the right track. Now here's a short program that illustrates the principle perfectly.

The Dreaded Telephone List!

Yes it is; once again I've got a telephone list. This is by Chris Oliver of Barton-under-Needwood, and took my eye because it illustrated beautifully random access filing. It was clear, it had clear accompanying noted *and* it came with a disc. So let's look at the program. Chris uses two random access files, a main record file holding names, addresses and numbers, and a file of key numbers, used as an index. The index number-keys must be filed as string arrays, dimensioned so that, as in all random access files, each entry in a particular file is the same length as the others.

Listing 1
This program creates and reserves space on the disc for two files. Line 2 creates a random access file with a record length of four: that is three digits, plus one for the ENTER used as a separator and automatically inserted each time we print # to the file. This gives 999 of these records plus an extra two at the end which are used as slots for storing counters. Line 3 creates a further file of record lenght 63 (plus one for the separator): because −1 is stipulated for the file size, it uses *one half of the space available on the disc*, which is why the index file was created first. Next, the positions used for storing the counters are primed (lines 5-8). As Chris says, this is a good illustration of the method Discovery uses to write to a random access file. This part of the program is then no longer needed.

Listing 2
The first thing this program must do is fetch the current values of the counters before manipulating or adding further records, and this is done in lines 70-90 which illustrate clearly the method of reading from a random access file. And this now covers the basis of the program; the rest is merely manipulation.

As Chris says, there are one or two things mentioned only briefly in the Opus manual which are worth emphasizing. Firstly, you cannot get a stream to flow in two ways at once. Since there are plenty of streams available, Chris finds it sensible to reserve some for writing to and others for reading from files. Chris also warns, too, of using **CLEAR** #: whilst useful on occasions for closing all streams, it does what it says. Thus, it clears out everything in all open streams, doing a Paul Daniels on the record you were trying to send to the file. So in routine "find", line 690, **CLEAR** # could be used, but in the "delete record" routine which moves key-numbers in the index file, each stream must be closed (line 760) to send the latest r$ to the file.

The manual does not mention that if you **POINT** # **n;x** and "x" is not a valid number, you get the "nonsense in Basic" report. And unless you are a *very* fast reader, you will need to control the scrolling when printing the contents of a file to the screen. A simple variable is used in the "list-all" and "scroll" routines of the example program.

Improvements

The example program has been deliberately kept simple. Improvements could include use of subroutines for reading from or writing to random access files and use of an "inserting search" routine instead of the simple progressive search could speed things up. In addition, the crude "change record" routine could be replaced with an editing routine, and an "Add-More?" option in the add item routine would be helpful. Use of machine code insert and find routines and/or the RAM disc (particularly the big 128K one) would also make things run faster. Finally, I had a glitch in the discs (which got an attack of perpetual motion) or something while the program was running: all innocent, I RAN it again. That loses the files; this could be avoided by having listing 1 as an autorunning subroutine

Listing 1

```
1 REM  First create the two Random-Access Files:- Main file "directory" holds
  names and phone numbers; each record can total up to 63 characters;
  Index-file "index" to hold numbers up to three digits long i.e. 1 to 999
  PLUS separators !
2 OPEN #6;"m";1;"index"RND4,1001: CLOSE #6
3 OPEN #4;"m";1;"directory"RND64,-1: CLOSE #4
4 REM  The last two positions in the index file ( 1000 & 1001 ) are reserved
  for counters (not dalmations) and need priming with Zeroes.
5 OPEN #7;"m";1;"index"RND4
6 POINT #7;1000: PRINT #7;"000"
7 POINT #7;1001: PRINT #7;"000": CLOSE #7
8 CLOSE #7: STOP
9 REM ****************************************************************
```

jumped over in later program repeats. Nevertheless, the program works, works satisfactorily and illustrates well random access files.

An Alternative

Chris writes that he has used machine code to reduce the six minutes of John Bennett's BASIC screen dump, but does not know how to pass non-characters to the printer without returning to a BASIC LPRINT statement. He would also like to know the addresses of the ports used and how to initialise them. Can anyone help him?

Several other readers' comments might also be of interest. For instance, David Calow of Worksop mentions that if he SAVEs a program to disc as follows:

SAVE * 1;"name??" LINE 10

and when he gets to the question marks, changes the ink to 7 and adds, instead, a couple of letters before changing the ink back to 0, then this protects the program. He says that you must have line 10 autorun or the computer won't accept it. If you try and load by typing in the visible name after a CAT, you merely get the "file not found" error. It also prevents accidental erasure, perhaps even more important. Oh, and if you suffer from amnesia, then just change paper to another colour before you CAT, and all is revealed.

RAM Chip

A query from a computer club concerned 5.25 inch drives. Is it possible to run *two* of these drives from a Discovery unit, and what is the RAM chip? The answr is that the RAM chip is pretty standard and not difficult to get; it is an IC 6116 chip, and yes, you can run two 5.25 inch drives if you so wish; don't forget to unplug the 3.5" original Discovery drive from the printed circuit board. Mr D. D. M. Cherry of Fort Gale, Umtata, Rep. of Transkei, asks similar questions and also asks about disc drive specifications. Almost any modern disc drive will do, but it needs to be Schugart-compatible and you need to add a power supply unit if it hasn't got one.

Errors

B. Domb of Wettingen, Switzerland, mentioned that he had read my comments about disc I/O errors in the December issue. He had had continual errors of this sort, and noticed that they often appeared through interference from other peripherals, for instance switching off a printer. He recommends that if you are

Listing 2

```
  1Ø CLS : PRINT "LOADING ""TELEPHONE DIRECTORY"""
  2Ø DEF FN C$(i$)=CHR$ (CODE i$-(32*(CODE i$>9Ø))): REM Fool the computer into
     thinking that you are entering CAPITALS, whether you are or not.
  3Ø DIM f$(63): DIM r$(3): REM for 1 to 999 records, up to 63 characters each.
  4Ø LET retmen=38Ø: LET find=6ØØ: LET insert=9ØØ: LET limit=999: LET duff=21ØØ
  5Ø GO SUB 1ØØØ: REM load message strings.
  69 REM  Fetch record-total and end-of-file values.
    "rectot" (stored at 1ØØØ on the index file) = No. of valid records in index.
    "filend" (stored at 1ØØ1 on index file) = No. of records on the file, includ
ing deleted ones!
  7Ø OPEN #6;"m";1;"index"RND4
  8Ø POINT #6;1ØØØ: INPUT #6;r$: LET rectot=VAL r$
  9Ø POINT #6;1ØØ1: INPUT #6;r$: LET filend=VAL r$: CLOSE #6
  95 IF rectot=Ø THEN CLS : PRINT "! New file !"''"No records yet entered."''x$:
 GO TO 21Ø
  99 REM
 1ØØ REM  CONTROL LOOP
 1Ø1 REM  ------------
 11Ø CLS : GO SUB 8ØØ: REM Print-menu
 12Ø IF INKEY$="" THEN GO TO 12Ø
 13Ø LET i$=FN C$(INKEY$)
 14Ø IF i$="A" THEN GO TO 2ØØ
 15Ø IF i$="L" THEN GO TO 3ØØ
 16Ø IF i$="F" THEN GO TO 4ØØ
 17Ø IF i$="C" THEN GO TO 5ØØ
 18Ø IF i$="D" THEN GO TO 7ØØ
 185 IF i$="Q" THEN STOP
 19Ø GO TO 12Ø
 199 REM
 2ØØ REM  ADD-ITEM ROUTINE
 2Ø1 REM  ----------------
 2Ø5 CLS : PRINT AT 5,2;x$
 21Ø DIM f$(63): INPUT "Enter Name : ";t$'"Phone No.     ";n$:
REM :" DIM " clears f$ for new input
 212 LET lt=LEN t$: LET ln=LEN n$: IF lt<1 OR ln<1 THEN GO TO 21Ø
 214 IF lt>31 OR ln>21 THEN GO TO duff
 216 IF CODE t$(1)<65 OR CODE t$(1)>122 OR n$(1)=" " THEN LET error=1: GO TO duf
f
 218 FOR n=1 TO lt: LET f$(n)=t$(n): NEXT n
 22Ø IF CODE n$(1)<47 OR CODE n$(1)>58 OR n$(1)=" " THEN LET error=2: GO TO duff
 225 CLS : PRINT s$: REM Saving message.
 23Ø FOR n=42 TO 41+LEN n$: LET f$(n)=n$(n-41): NEXT n
 24Ø LET f$(33 TO 41)="Tel. No. "
 25Ø LET q$=f$: LET q$(lt+1 TO lt+1)="$": LET q$(ln+42 TO ln+42)="$"
 26Ø GO SUB insert
 265 OPEN #5;"m";1;"directory"RND64: REM  Using streams 4 & 6 for reading FROM
    files and odd numbered streams 5 & 7 for writing TO files.
 27Ø POINT #5;filend: PRINT #5;f$
 275 CLOSE #5
 28Ø OPEN #7;"m";1;"index"RND4
 285 POINT #7;1ØØØ: PRINT #7;STR$ rectot: REM Update pointers
 29Ø POINT #7;1ØØ1: PRINT #7;STR$ filend: REM   see Line 69
 295 CLOSE #7: GO TO 11Ø
 299 REM
 3ØØ REM  LIST-ALL ROUTINE
 3Ø1 REM  ----------------
 3Ø5 CLS : LET scroll=7: REM Need to control scrolling or whole file is dumped.
 31Ø OPEN #4;"m";1;"directory"RND64: OPEN #6;"m";1;"index"RND4
 32Ø LET pntr=1
 33Ø POINT #6;pntr: INPUT #6;r$: POINT #4;VAL r$
 335 IF pntr>scroll THEN GO SUB 15ØØ
 34Ø INPUT #4;f$
 35Ø PRINT f$''
 355 LET pntr=pntr+1
 36Ø IF pntr<=rectot THEN GO TO 33Ø
 37Ø CLEAR #
 379 REM  RETURN TO MENU SUBroutine "retmen"
 38Ø PRINT #1;"""M"" for Menu"
 39Ø IF FN C$(INKEY$)<>"M" THEN GO TO 39Ø
 395 GO TO 11Ø
 399 REM
 4ØØ REM  SEARCH ROUTINES
 4Ø1 REM  ---------------
 41Ø INPUT AT 22,Ø;AT 5,Ø;"Please type in name OR number"'"that you are looking
for ? "'';i$
 42Ø LET pntr=1: GO SUB find
 44Ø LET pntr=pntr+1
 45Ø IF pntr<rectot+1 THEN GO SUB find+25: GO TO 44Ø
 46Ø IF found=Ø THEN CLS : PRINT k$;a$'' BRIGHT 1;i$: REM kan't find message
 47Ø PRINT ''e$: REM end of search message.
 48Ø GO TO retmen
 499 REM
 5ØØ REM  CHANGE-RECORD ROUTINE
 5Ø1 REM  ---------------------
 5Ø5 LET d$="Change": INPUT AT 22,Ø;AT 5,Ø;"Please type in name OR number"'"that
 you wish to change ? "'';i$
 51Ø LET pntr=1: GO SUB find
 515 IF found=Ø THEN GO TO 535
 517 PRINT AT 5,Ø;"Do you wish to:-"''TAB 14;d$;" this ?"''TAB 14;"Look for simi
lar ?"
 52Ø LET w$=FN c$(INKEY$)
 525 IF w$<>"C" AND w$<>"D" AND w$<>"L" THEN GO TO 52Ø
 53Ø IF w$<>"L" THEN GO TO 72Ø: REM ...so slip into the Delete routine!
 532 LET found=Ø
 535 LET pntr=pntr+1
 55Ø IF a$="another " THEN CLS : PRINT BRIGHT 1;l$;a$;i$
 555 IF pntr>rectot THEN GO TO 46Ø: REM end of search
 56Ø GO SUB find+25
 57Ø GO TO 515
 599 REM
 6ØØ REM  SUBroutine "find"
 6Ø1 REM  ---------------
 61Ø CLS : LET offset=Ø: LET found=Ø
 615 PRINT BRIGHT 1;l$;i$: REM looking . . . message
 619 REM test char/num Input
 62Ø IF i$(1)<":" THEN LET offset=41
 625 OPEN #4;"m";1;"directory"RND64: OPEN #6;"m";1;"index"RND4
 63Ø POINT #6;pntr: INPUT #6;r$: POINT #4;VAL r$
 635 INPUT #4;f$
 64Ø FOR q=1 TO LEN i$
 645 IF FN c$(f$(q+offset))<>FN c$(i$(q)) THEN CLEAR #: RETURN
 65Ø NEXT q
 66Ø LET found=1: LET a$="another "
```

```
 670 PRINT 'f$''
 690 CLEAR #: RETURN
 699 REM
 700 REM  DELETE-RECORD ROUTINE
 701 REM  ~~~~~~~~~~~~~~~~~~~~~
 705 LET d$="Delete"
 710 INPUT AT 22,0;AT 5,0;"Please type in name OR number"'"that you wish to Dele
te ? "'';i$
 715 GO TO 510: REM go and look for it first!
 720 OPEN #6;"m";1;"index"RND4: OPEN #7;"m";1;"index"RND4
 725 LET rectot=rectot-1
 730 FOR x=pntr TO rectot
 740 POINT #6;x+1: INPUT #6;r$: POINT #7;x: PRINT #7;r$
 750 NEXT x
 755 POINT #7;1000: PRINT #7;STR$ rectot: REM remember to store adjusted value.
 760 CLOSE #7: CLOSE #6: REM Don't use CLEAR # here!!
 770 IF w$="C" THEN CLS : PRINT f$;AT 17,0;o$: GO TO 210: REM old record deleted
     so add new version.
 780 CLS : PRINT z$: REM zapped message !
 790 GO TO retmen
 799 REM
 800 REM  PRINT-MENU SUBroutine
 801 REM  ~~~~~~~~~~~~~~~~~~~~~
 810 IF filend=limit THEN PRINT AT 10,10;"FILE FULL": STOP
 820 PRINT AT 2,8;" OPTIONS "
 830 PRINT AT 5,0;"Add a new entry."''"Change an existing entry."''"List all (in
alphabetical order)"''"Find a name OR number"''"Delete an entry."
 840 PRINT #1; INK 1;"  Records on File : ";rectot'"Space remains for : ";(limit
-filend)
 860 LET a$="": REM reset var "another"
 890 RETURN
 899 REM
 900 REM  SUBroutine "insert" : REM Adjusts the index so that records appear
 901 REM  ~~~~~~~~~~~~~~~~~~         to be filed in alphabetical order, so
                                    reducing recall time when listing.
 905 OPEN #4;"m";1;"directory"RND64   : REM read from main file
 910 LET pntr=rectot: LET rectot=rectot+1: IF rectot=1 THEN GO TO 950
 915 OPEN #6;"m";1;"index"RND4        : REM read from index file
 917 POINT #6;pntr: INPUT #6;r$
 918 CLOSE #6
 920 POINT #4;VAL r$: INPUT #4;g$
 925 LET el=1: REM pointer to element within arrays & file records.
 930 IF FN c$(q$(el))>FN c$(g$(el)) THEN GO TO 950
 935 IF FN c$(q$(el))=FN c$(g$(el)) THEN GO TO 970
 940 OPEN #7;"m";1;"index"RND4        : REM write to index file
 942 POINT #7;pntr+1: PRINT #7;r$
 943 CLOSE #7
 945 LET pntr=pntr-1: IF pntr>0 THEN GO TO 915
 950 LET filend=filend+1
 955 OPEN #7;"m";1;"index"RND4        : REM write to index file
 957 POINT #7;pntr+1: LET r$=STR$ filend: PRINT #7;r$
 958 CLOSE #7
 960 CLOSE #4: RETURN
 970 LET el=el+1
 975 IF q$(el)<>"$" THEN GO TO 930
 980 LET rectot=rectot-1
 985 CLOSE #4
 990 GO TO 2000: REM input same as a record on file !
 999 REM
1000 REM      MESSAGES
1001 REM      ~~~~~~~~
1020 LET k$="SORRY -  Can't find "
1030 LET e$="******** END OF SEARCH *********"
1040 LET l$="Looking for "
1050 LET z$="Zap ! - Entry now Deleted."
1060 LET s$="Storing new information
     Please wait !"
1070 LET o$="Old entry above is now deleted. Type in the new !"
1090 LET x$="Add new entry:-"
1095 RETURN
1099 REM
1500 REM  SCROLL SUBroutine
1501 REM  ~~~~~~~~~~~~~~~~~
1510 PRINT #1;"Scroll ?": PAUSE 0
1520 IF FN c$(INKEY$)="N" THEN CLOSE #4: GO TO 110
1530 LET scroll=scroll+7
1550 RETURN
1599 REM
2000 REM  IDENTICAL RECORDS !?*
2001 REM  ~~~~~~~~~~~~~~~~~
2010 CLS : PRINT g$''"         Already on file !!"
2020 CLOSE #4
2030 GO TO retmen
2099 REM
2100 REM  ERRONEOUS (duff) ENTRY
2101 REM  ~~~~~~~~~~~~~~~~~~~~~~
2110 CLS : PRINT AT 8,3;"Sorry !   Entry Error -"
2120 IF lt>31 THEN PRINT ''t$'"Too long.": GO TO 2020
2130 IF ln>21 THEN PRINT ''n$'"Too long.": GO TO 2020
2140 IF error=1 THEN PRINT ''"Please type NAME first": GO TO retmen
2150 IF error=2 THEN PRINT ''"Please use numbers !": GO TO retmen
2190 GO TO 2020
```

plagued like he was, and sending your Discovery to Opus merely costs you postage but does not alleviate the problem, try switching on Discovery *last*.

He also mentions that he can often get rid of this error by FORMATting an empty disc, but warns you to make sure that it is empty, because "destroy?" will

Figure 1. UDG Search and replace.

```
  10 CLEAR USR "a"-61: LET a=USR
 "a"-60: PRINT "Code located at
address "'a''"Use by  RANDOMIZE
USR "'a
  20 FOR i=a TO a+55: READ a$: P
OKE i,((CODE a$(1)-48-(7 AND a$(
1)>"9"))*16+(CODE a$(2)-48-(7 A
ND a$(2)>"9")): NEXT i
 100 DATA "ED","4B","4B","5C","2
A","53","5C","18","17","7E","FE"
,"0E","20","10","FE","0D","28","
0D","FE","22","28","0F","23","A7
","ED","42","D0","09","18","EB"
 110 DATA "23","23","23","23","2
3","18","F1","23","7E","FE","22"
,"28","EB","D6","90","38","F6","
D6","15","30","F2","C6","56","77
","18","ED"
```

not appear, and valuable information could be lost.

Alphacom problem

Sgt McMenzie of BFPO 42 says he has trouble with an Alphacom printer and Discovery, and asks if this is due to Discovery's port being non-standard. The answer is that the Alphacom is non-standard, some models more than others, but that they all contain their own comic way of interpreting the signals from the Spectrum and the Discovery interferes with this. So, you can't use both effectively together.

Figure 2.

```
  1 REM AMSTRAD DMP 2000:               graphic dumper
 10 REM first, load screen                  for dump
350 OPEN #3;"B"
360 LET l$=CHR$ 27+"J"+CHR$ 20
370 INPUT "double strike? Y/N:";S$
380 LET st=1+(s$="Y")+(S$="y")
400 INPUT "width (1 to 7) ";w
410 INPUT "height (1,2,4,8)";h
420 POKE 23259,h
430 POKE 23258,w
440 FOR f=175 TO 0 STEP -8/h
450 LPRINT l$+CHR$ 13;
460 FOR t=1 TO st
470 LPRINT CHR$ 27+"J"+CHR$ 2+CHR$ 13;
480 POKE 23261,f
490 LPRINT CHR$ 27+"Z"+CHR$ 0+CHR$ w;
500 RANDOMIZE USR 60245
510 NEXT t
520 NEXT f
530 STOP
560 SAVE *1;"dumper" LINE 9999
570 SAVE *1;"dumperCODE "CODE 60245,93
580 STOP
600 CLEAR 60244: LOAD *1;"dumperCODE "CODE 60245,93
```

LLISTing UDGs

An anonymous reader from New Milton, Hants, mentions that Discovery's printer port has an irritating habit, namely that of sending question marks instead of user-defined graphics: he points out that a program by our own Ray Elder (ZX, Apr/May 1984, p53), originally written for the old Kempston interface works equally well with Discovery. Here it is, again (Figure 1); it merely replacs all the user defined graphics with appropriate letters, A to U.

Dump to an Amstrad

Tim Andrews of Bromley has sent in a program to send a dump to an Amstrad DMP 2000, In the BASIC listing (Figure 2), line 360 sets up the line feed, and line 470 increments this by a small amount. Line 490 sets the printer for quadruple density graphics and also to prepare to print 256*w the number of graphic lines. The code is shown in Figure 3. Tim says that due to his relative inexperience, he only used the BC, HL and A registers; it could perhaps have been shorter otherwise. He also asks about peripherals, being scared to use his SPECDRUM in case everything blew up. It should be

all right, particularly if one is using a 48K Spectrum, as the power supply is ample even for a 128 and most peripherals, provided that there is no clash, particularly over the NMI line which is not continued through the Opus port.

Auto-run select

And last, I shall break my rule of not publishing any more CAT routines, because this is one with a difference. You remember Mr. Nutting's program, Supercat, in last October's ZX Computing? Well, by using his "catc" CODE and modifying the BASIC, Alan Harper has made a loader so simple that even his child of five can use it. The BASIC, listed in Figure 4, is saved as "run" LINE 9999. When run, the program automatically lists all the BASIC programs on the disc and assigns a number to each: entering the number of the required program then loads it. Neat and effective.

Tailpiece

Opus have clearly abandoned the Discovery, but we will continue to keep this column going as long as there is sufficient interest. Keep the programs coming. See you next month.

Figure 3. Assembly listing for screen dumps.

```
60245 21DC5A   LD   HL,23260
60248 3600     LD   (HL),000
60250 21DE5A   LD   HL,23262
60253 3600     LD   (HL),000
60255 3E01     LD   A,001
60257 21DB5A   LD   HL,23259
60260 46       LD   B,(HL)
60261 0E10     LD   C,016
60263 CB09     RRC  C
60265 B8       CP   B
60266 2804     JR   Z,60272
60268 CB08     RRC  B
60270 18F7     JR   60263
60272 41       LD   B,C
60273 C5       PUSH BC
60274 21DD5A   LD   HL,23261
60277 7E       LD   A,(HL)
60278 90       SUB  B
60279 47       LD   B,A
60280 21DC5A   LD   HL,23260
60283 4E       LD   C,(HL)
60284 CDAA22   CALL 08874
60287 47       LD   B,A
60288 04       INC  B
60289 7E       LD   A,(HL)
60290 07       RLCA
60291 10FD     DJNZ 60290
60293 E601     AND  001
60295 21DB5A   LD   HL,23259
60298 46       LD   B,(HL)
60299 21DE5A   LD   HL,23262
60302 F5       PUSH AF
60303 86       ADD  A,(HL)
60304 77       LD   (HL),A
60305 CB0E     RRC  (HL)
60307 F1       POP  AF
60308 10F8     DJNZ 60302
60310 C1       POP  BC
60311 10DB     DJNZ 60273
60313 3E03     LD   A,003
60315 CD0116   CALL 05633
60318 21DA5A   LD   HL,23258
60321 46       LD   B,(HL)
60322 C5       PUSH BC
60323 3ADE5A   LD   A,(23262)
60326 D7       RST  016
60327 C1       POP  BC
60328 10F8     DJNZ 60322
60330 21DC5A   LD   HL,23260
60333 34       INC  (HL)
60334 20AA     JR   NZ,60250
60336 C9       RET
```

Figure 4. Auto-run program.

```
   1 REM
   2 REM  AUTO RUN SELECT
   3 REM
   4 REM  BY  ALAN HARPER
   5 REM
   6 DIM n$(50,10)
  10 LET d=1: POKE 64990,d
  20 RANDOMIZE USR 64900
  30 DEF FN a(x)=PEEK (x+a)+256*
PEEK (a+x+1)
  40 CLS : PRINT AT 10,9;"PLEASE
 WAIT "
  50 LET a=62016: LET i=63800
  55 LET n=1
  80 LET e=FN a(4)
  95 LET m$=""
 100 IF e=65535 THEN GO TO 300
 110 FOR p=a+6 TO a+15: LET m$=m
$+CHR$ PEEK p: NEXT p
 130 LET t=a: LET a=i: LET type=
PEEK a
 140 IF m$(1 TO 3)="run" THEN GO
 TO 220
 170 IF type=0 THEN LET n$(n)=m$
: LET n=n+1
 220 LET a=t: LET a=a+16: LET i=
i+10: GO TO 60
 300 CLS : PRINT "DISC NAME =";:
 FOR a=62006 TO 62015: PRINT BRI
GHT 1;CHR$ PEEK a;: NEXT a: PRIN
T : PRINT : PRINT
 305 FOR q=1 TO n-1
 310 PRINT PAPER 1; INK 6;q; PAP
ER 7; INK 0;"= ";n$(q),
 320 NEXT q
 330 INPUT "RUN which one ? ";u
 331 IF u>(n-1) THEN GO TO 330
 335 CLS
 340 LOAD *1;n$(u)
9000 STOP
9100 SAVE *1;"catc"CODE 64900,10
0
9110 SAVE *1;"run" LINE 9999
9998 STOP
9999 CLEAR 61999: PRINT "MACHINE
 CODE IS LOADING": LOAD *1;"catc
"CODE : GO TO 1
```

IN-FLIGHT

Tony Hetherington takes a spin in four of the latest Spectrum flight simulators.

TOP GUN
Ocean
£7.95

Games based on top films have often been top flops but this high flyer is set to reverse the trend and is soaring up the chart as a game in its own right.

You star in the game that pits you in a high speed aerial duel against either a human opponent or a series of computer opponents. The action takes place on a vertical split screen display with vector graphics representing the F14 Tomcat fighters. I found the vertical split screen displays worked exceptionally well and made the game a lot easier than the normal horizontal display (one player's screen on top of the other's), particularly for players hunched either side of the keyboard. You can of course use joysticks if you have them but single stick owners may opt for the keyboard to make the battles fair. If you're up against the computer you'll need all the help you can get.

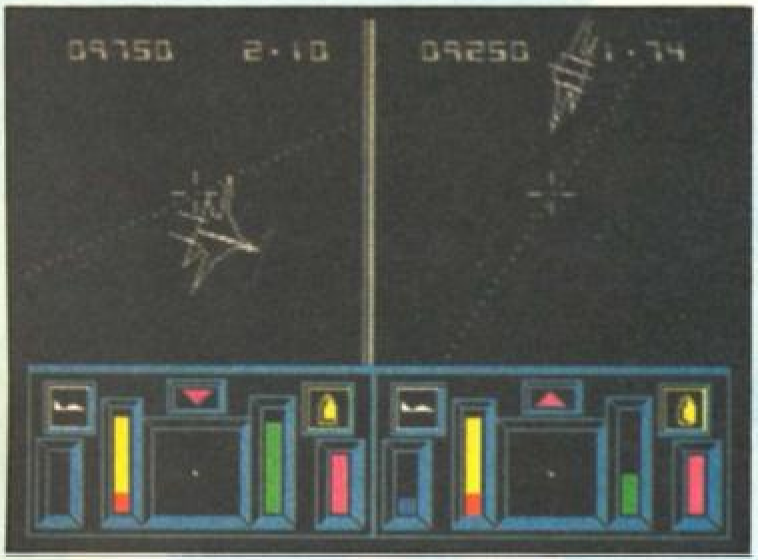
Control of the F14s is a mixture of the standard climb, bank and dive controls with additional keys to control the throttle and change weapons while reading the instruments that report your speed, altitude, attitude (climbing or descending) and plot your opponents position on your radar. Luckily, these displays are easy to read even when travelling at speed as that's the way this game's played.

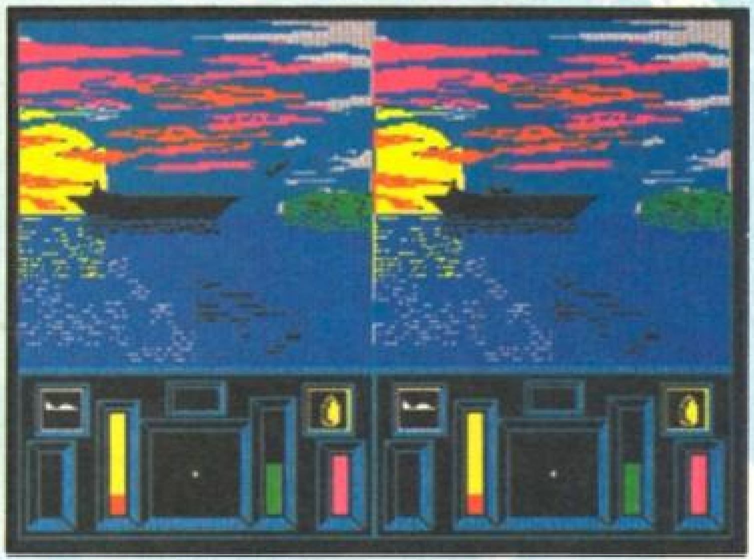
The contest begins as both fighters take-off from their respective carriers and head straight for each other. Thanks to the simple flight controls you can concentrate on the combat and decisions on whether to attack with your machine guns or fire a sidewinder homing missile. A missile will wipe out your opponent in a single shot but you will have to keep him in your sights while your missile systems lock on. To add to your problems and also your chances of survival you can also fire a flare to distract enemy fire but since you can only select one system at a time you have to anticipate your opponent's actions almost before he knows them himself. Even the simple machine guns can cause you problems as they can overheat and jam before you can score enough hits to destroy him.

The two player game is a simple head to head contest where both players have a stock of three planes. The last plane flying wins the contest.

Against the computer your three planes (or lives) must tackle a series of computer pilots that increase in skill. You don't score any points, just survive to fight the next level or group of three fighters. The first two levels are relatively easy as the enemy mainly attacks with it's machine guns and fall easy prey to your sidewinders but from then on it gets a lot harder as you face a succession of missile firing goes. To win you will really have to be Top Gun.

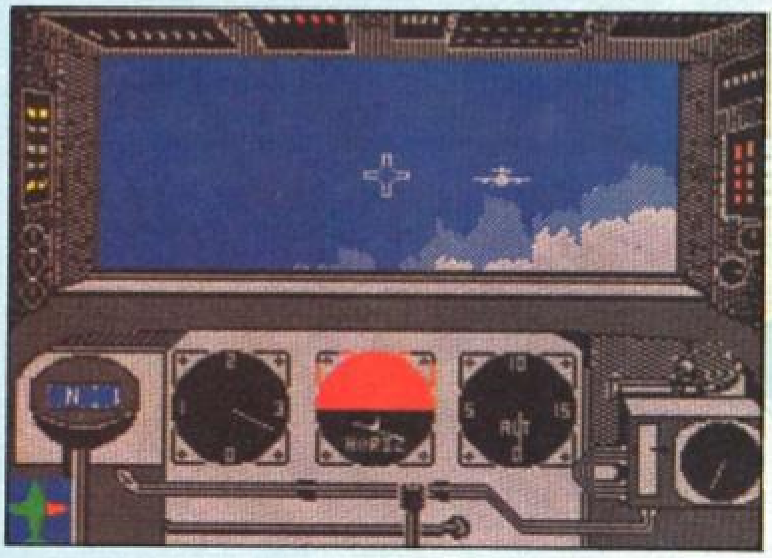


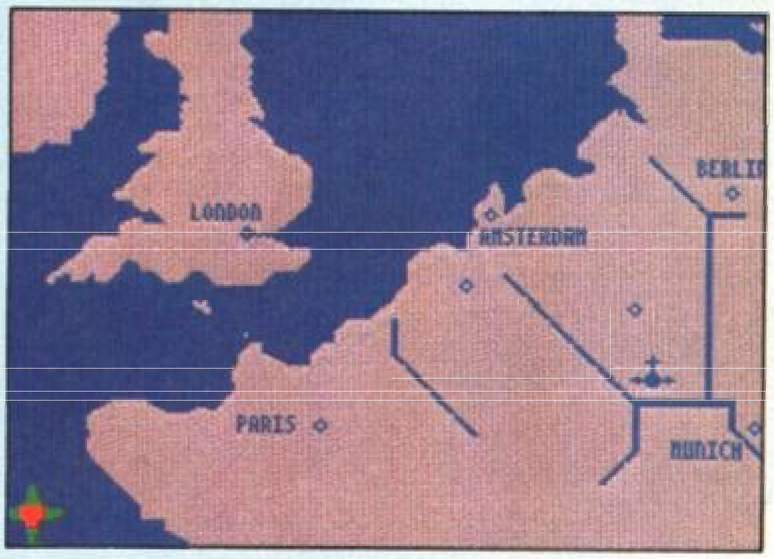




TOP GUN

ACE OF ACES
US Gold
£9.95





ACE OF ACES

"No British aircraft will ever bomb Berlin" was the claim of Luftwaffe chief Herman Goering. Now, thanks to US Gold, you can join the ranks of RAF Mosquito pilots that not only bombed Berlin but also trounced trains, pummelled planes, sank submarines and even scored victories against the dreaded V-1 buzz bombs.

Your game begins in the briefing room with your possible targets marked on the Squadron Leader's blackboard. You choose whichever ones you like, go on a practise mission or attempt the Ace of Aces grandslam and go for the lot. However before you can bomb submarines and trains while fighting Me109s, London bound bombers and V-1s you'll need to learn how to fly this highly versatile fighter bomber.

The mechanics of flying are similar to those experienced by Lancaster bomber pilots in the excellent Dambusters game and consists of a series of screens each with their own set of controls. The plane is steered from the pilot's screen that shows the view through the front of the aircraft as well as instruments to monitor the plane's height, speed and course and direction and height of any enemy "bandits". The Mosquito has two Rolls Royce engines that are controlled directly from the port and starboard engineer controls. These monitor the throttle and boost given to each engine and also contain a fire extinguisher in case the engines overheat. There are also flaps and landing gear controls that can be used in evasive tactics as they'll cause a sudden loss of speed although you don't need them to land or take-off as the computer does that for you.

On course

Selecting the navigator's screen punches up the mission's tactical map that plots the relative positons of you, the target(s), any bad weather that can make flying even more hazardous and, of course, the dangerous Me109 fighters. Once you reach your target (on a bombing mission) it's all up to the skill, accuracy and timing of the bombardier. Opening the bomb doors will hopefully reveal the target that can be destroyed by your bombs. This screen is also used to dump empty fuel tanks to lighten the load and to switch the forward gunner (pilot screen) from cannon to rockets.

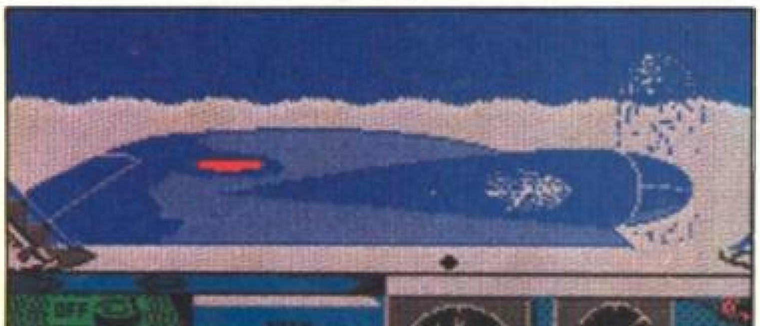
Naturally each mission is different and requires specialist tools, so you can customise your weapons load before you set off adding extra bombs, rockets or cannon ammunition and extra fuel tanks for longer flights.

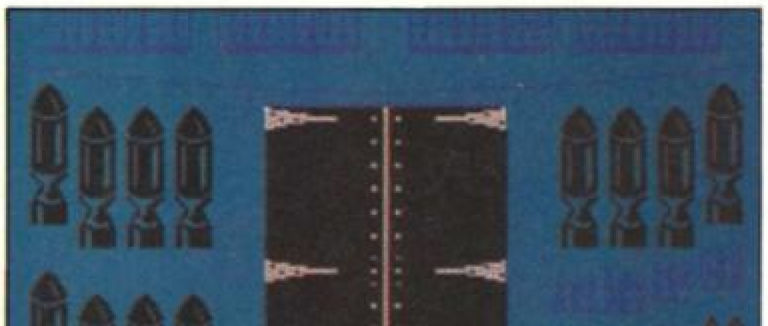
To finish your training you should fly a few practise missions to tune up your timing as you bomb the train (taking care to hit only the carriages that don't contain POWs) and submarines before they dive to safety.

In the real bombing mission it is vital that you get everything right as you only get one chance. The ideal approach is at 100mph at anything under 1,000 feet but this might not be always possible if you've been damaged by Me109 attack.

After the raid get back to blighty in time for another mission. Your performance is based on a points system with high scores awarded for getting back as well as for targets hit and even bombs, fuel and rockets saved.

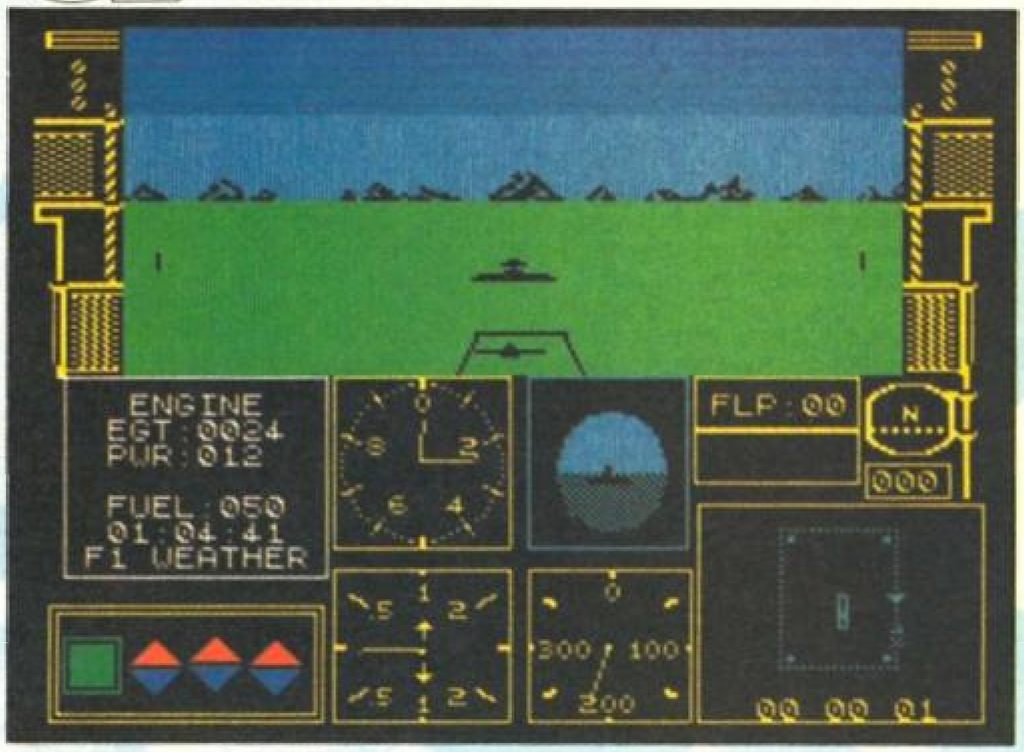
After a few successful sorties you'll be ready for a double mission, perhaps defending London against V-1 and bomber attacks until finally you go for the grandslam and become one of the Ace of Aces.





AcroJet

ACROJET
Microprose
£9.95



More high flying antics as man and machine are again driven to their limits, only this time it's a battle against the clock since you're competing in an airbourne decathlon — ten events that require supreme control of your BD5-J 200mph jet aircraft if you are to take the championship and enter the hall of fame. Before you write your name into the record books you'll have to learn how to fly.

This is undoubtedly the most difficult of this month's simulators to keep off the ground let alone trying to compete in the acrobatic events. This isn't made any easier when you consider that the instructions were written for the Commodore 64 with a brief and extremely vague errata sheet to explain the Spectrum keys! "Use joystick with Interface 2 or keys 6,7,8,9,0 for controlling plane" guarantees initial disaster flights as you sort out the controls, and it also rules out the most popular joystick interfaces.

The screen display is custom built for sport flying and contains a bank of instruments and a 3D view that also includes your aircraft as it swoops and banks around the circuit. A plan of the course which also doubles as a radar combines with a compass to keep you on course as you navigate your way around the hazards.

The first event is a relatively simple task to circle four pylons before returning to the landing/take-off strip. In the Slalom race things are a little trickier as you have to circle the same pylons but in a set order and in the correct direction.

If flying is still more luck than judgement then you should quit now and be grateful for the points you've already got because the next event involves cutting a 3" ribbon strung between two poles. To show that wasn't a fluke you have to cut more ribbons while flying upside down, then in the ribbon roll you must cut one ribbon, roll over and cut another without catching your wings on the ground. If that was too easy why not try the under ribbon low level race, the loop and cuban eight which contains everything you've risked so far and the insane spot landing in which you must deliberately stall your engine at 2000ft then land with pinpoint accuracy. If you turn on your engine you lose points but if you don't you might lose your plane.

Luckily you're awarded points even if you don't finish the course so even beginners can compete with each other.

Up to now I've perfected my own events including the crash-at-the-end-of-the-runway, the smash-into-the-pylon and freestyle-plummet. It'll take a lot more practise before I can take on Microprose's own Major Bill Stealey.



DEEP STRIKE
Durell
£9.95

World War I dogfights were extremely hazardous with planes often shooting off their own propellers in the heat of the action. Add to that the unreliability of such flimsy aircraft and you realise the danger of the mission you've been sent on. Your mission is to fight off hordes of Red Barons, Black Barons and Blue Maxes to protect the bombers that fly in front of you.

This aerial convoy has its own problems in that you have to drop the bombs while trying not to shoot the bomber that spends most of the game in front of your gunsights!

The screen shows your view over the scrolling landscape and your controls that are dominated by two propellers which show the deteriorating condition of your plane and your bomber, and a pointer that keeps you on course.

The best results seem to be achieved if you fly so that you fire just below the bomber's undercarriage but be prepared to move out to intercept enemy fighters and gun down barrage balloons before they destroy your bomber. As each bomber is destroyed another takes its place until you either get a chance to bomb your main target (the enemy fuel dump) or you're left on your own to take out as many of their planes as you can before they get you.





MEMORY EXPANSION PROGRAM

Toni Baker shows how to squeeze some extra memory out of 16 and 48K Spectrums.

PROGRAM

```
 10 CLEAR 32767
 20 LET X=32768
 30 FOR I=0 TO 14
 40 LET C=0
 50 FOR J=1 TO 6
 60 READ A
 70 POKE X,A
 80 LET X=X+1
 90 LET C=C+A
100 NEXT J
110 READ A
120 IF A<>C THEN  PRINT "ERROR
N LINE ";10*I+160: STOP
130 NEXT I
140 RANDOMIZE USR 32786
150 STOP
160 DATA 22,0,0,17,7,16,62
170 DATA 3,65,80,82,73,76,379
180 DATA 32,70,79,79,76,32,368
190 DATA 62,2,205,1,22,1,293
200 DATA 7,0,17,0,128,205,357
210 DATA 60,32,6,64,197,1,360
220 DATA 11,0,17,7,128,205,368
230 DATA 60,32,193,16,243,17,56

240 DATA 3,7,205,84,31,210,540
250 DATA 123,27,118,118,20,203,
09
260 DATA 154,28,203,155,213,122
875
270 DATA 205,155,34,230,248,179
1051
280 DATA 33,0,88,17,1,88,227
290 DATA 1,255,2,119,237,176,79

300 DATA 209,24,219,0,0,0,452
```

Once upon a time, when I was a lot younger than I am now, I was privileged enough to own a calculator. It wasn't anything flash, you understand, because calculators had only just come onto the market — but it was good enough for me at the time. It did add and subtract, multiply and divide. All very simple stuff, I grant you, but fun at the time.

A friend of mine also had a calculator. This was a more expensive version. It was made by the same company, but was the next one up in the range. It had an extra column of buttons on the right, and was coloured black instead of white. The extra buttons were things like M+, M−, MR and so on — all memory functions, allowing you to store and recall numbers and so on. Then, one day, while we were playing a casual game of throwing-calculators-around-the-room (as children do), the case of my prized calculator came apart, revealing a panel of little rubber buttons on one half, and a grid of wires on the other half. It became immediately apparent to me that pushing down one of the rubber buttons would make contact between a vertical wire and a horizontal wire, and hence register the key depression. It was then that I noticed an inconsistency — there were more vertical wires than there were columns of keys, giving an extra six possible contacts.

Surprise surprise!

Imagine my surprise when, with the case still open, I pressed on the extra contacts to see what happened — only to find that they performed the M+, M−, MR (etc) functions of the more expensive version. It seemed that the internal hardware of the two calculators was identical, but they were put into two different external cases so that they could sell at two different prices for differing sections of the market. I suppose it must have been cheaper for them to do that than to produce two entirely different machines.

With that little anecdote in mind, we return to the present. Calculators have long gone out of fashion — today, computers are in thing. I began to wonder, however, whether what was true of calculators in the early days, might also be true of computers now. Could it be possible that inside the various different boxes there was basically one, and only one, type of Spectrum?

Literally speaking, of course, this cannot be true. The IN ports seem to give different results for different issues of Spectrum, but the central problem remains. It is not possible that the advanced features of the Spectrum 128 (extra memory, full screen editor, and so on) are, and have always been, available on the old 48K and 16K Spectrums?

It turns out that is in fact the case. Both 16K and 48K machines, have in fact already got 128K of RAM built in — though accessing it has not been easy. On the Spectrum 128, outputting a byte to port 7FFD was sufficient to page in the extra memory. On the 48K and 16K machines this is not so.

It is possible, however, to devise a machine code program which will give *genuine* access to the extra

RAM and ROM memory. Such a program is included in this article. It is a BASIC program with the machine code held in DATA statements. The BASIC program — essentially a machine code loader, will POKE the machine code into memory and then, by means of a USR statement, run the machine code itself. The last item in each DATA statement is a checksum to ensure that the line has been typed in correctly. If the checksum does not match with the rest of the row then an error message is generated, indicating the line number at which the DATA was incorrect.

Once the program has been typed in, with each DATA statement error free, then the program may be RUN. There will be a short delay whilst the machine code is loaded into memory, and then the computer will, effectively, turn itself into a Spectrum 128, and will remain so until the power is switched off.

Reset

The effect of running this program is equivalent to pressing the RESET button on a normal Spectrum 128. The screen goes black with a white border for a few seconds before you are presented with the main 128 menu. A cyan bar outlines the first item on the menu, TAPE LOADER. Ignore this for the moment. Using the CURSOR DOWN function (CAPS-SHIFT/6 if you don't have a Spectrum+ with a separate CURSOR-DOWN key) move the cyan bar to the second item, 128 BASIC, and your machine is every bit as good as the 128 itself.

I would like to stress at this point that the program will convert a Spectrum to a Spectrum 128 *by software means alone!* This is important, because it means that the guarantee is not invalidated!

Of course the main problem with converting a Spectrum by this means, is that it doesn't supply an instruction book. It is not within the scope of this article to give detailed information on 128 BASIC, and furthermore, neither Sinclair nor Amstrad will provide an instruction book without supplying a computer as well. The only thing I can suggest is that you find a friend who's got a Spectrum 128, and borrow their instruction book. If you don't know anyone who owns a Spectrum 128 then it might be worth your while joining a local computer club. Most of the extra memory in a Spectrum 128 is locked up in a software data-storage system called silicon disc, or RAMdisc. It is worthwhile getting hold of an instruction book, for the simple reason that without it you won't know how to access the extra memory in 128 BASIC.

M/C

How the machine code program works is surprisingly simple. As has been previously stated, an output to port 7FFD alone is not sufficient. The Spectrum contains two new registers, which to my knowledge have never been properly exploited. They are referred to as "Alternative Programming Registers", and it is the first of these (APR1 for short) which discriminates between the various different hardware models of the Spectrum. Access to the chip is by a complex sequence of OUT instructions. This sequence is performed automatically by all Spectrums whenever the machine is either switched on or reset, so that the machine knows what kind of Spectrum it's supposed to be.

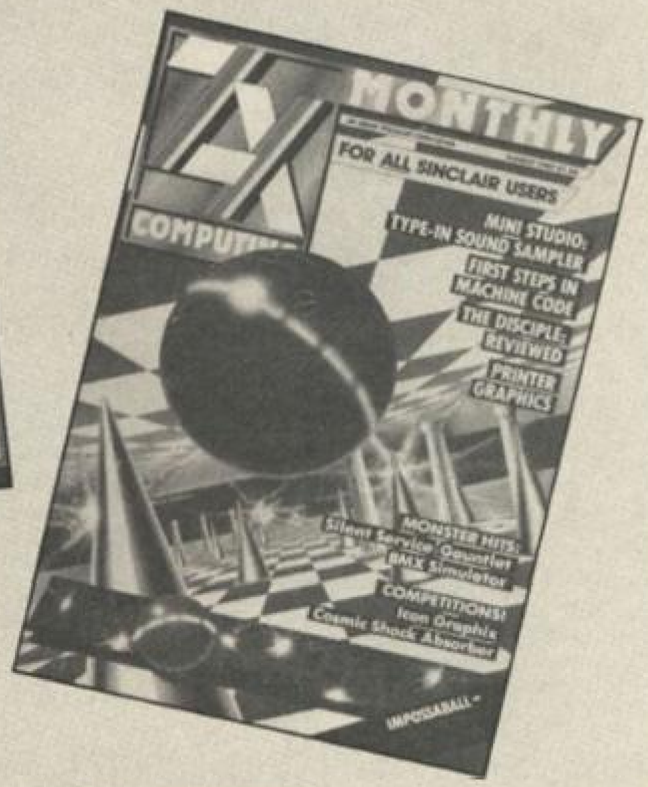
I'll leave it to you now to actually load in the program and try it out. Good luck, and I'll see you again next month.

ZX CONTACTS DIRECTORY

Who to call if you want your Spectrum repaired... the latest game... a new peripheral... it's all in this indispensable contacts guide







Before we start talking telephone numbers a brief word about the services which are available through ZX. We try to provide as full a service to our readers as possible and to get the most from us follow the guidelines that are outlined here.

Back Issues

Back issues are handled by a company called Infonet, Times House, 179 The Marlowes, Hemel Hempstead, Herts HP1 1BB. If you have missed an issue, copies may still be available from Infonet, price £1.50+50p towards postage and packing. Copies are still available for all ZX Computing Monthlies but copies of the former bi-monthly issues are stricly limited.
Infonet: 0442 48435

Subscriptions

All matters relating to subscriptions are handled by Infonet. If you want to subscribe at a special discount price (£3 off the usual annual sub price) refer to the subscriptions savings department at the end of this directory, but hurry as this offer ends on April 30th 1987.

Photocopies

If you want to get hold of an article from a past ZX there is a photocopy service available, price £1.50 per article. Send requests to the Golden Square address marking your envelope clearly "Photocopy service". Cheques payable to ASP Limited.

Binders

ZX Binders are available, price £5.20 inc p&p from ASP Readers Services, PO Box 35, Wolsey House, Wolsey Road, Hemel Hempstead, Herts HP2 4SS. 0442 41221. Cheques payable to ASP Limited.

Listing problems, etc...

Although we attempt to be as accurate as possible mistakes do periodically occur. If you have a problem with a listing or an article please *write* to us rather than phoning. If a mistake has been spotted then we will send you a correction sheet, if not we'll do our best to help. If you wish to correspond with a contributor over a problem with an article, write c/o ZX Computing, marking your envelope clearly with the contributors name. In all cases please enclose a SAE and mark your envelope clearly with an indication of it's content, ie the title of the listing. It's simply easier to process that way and you will receive a reply sooner. We try to be prompt in responding to letters but please understand if you suffer a delay in receiving a reply.

Future services

In the near future we hope to launch a Software Service offering ZX readers the chance to purchase tapes of series that have appeared in ZX. This is still in planning stage but if there are listings or series from ZX that you would particularly like to buy in tape form drop us a line, we'd like to hear from you — it could help to decide the first few releases.

Bryan Ralph, Editor,

ZX Computing Monthly, No1 Golden Square, London W1R 3AB (01 437 0626).

REPAIRS

When your Spectrum decides to go on the blink who do you turn to? ZX focuses on professional repair companies.

If you're one of the people who got a Spectrum this Christmas then it's quite likely that you're still at the stage where you're totally infatuated with the thing. Well it's time I put a stop to that.

It's one of those inevitable facts of life that your home computer, like any other household gadget, may one day blow a gasket and leave you in the lurch just when you need it. That's why we've listed all those repair companies in this issue. But just what are the frailties that the Spectrum is heir to?

We spoke to some repair services and it seems that all sorts of things have happened to machines at one time or another, from the fairly common cup of coffee getting knocked over them, to a TV aerial getting struck by lightning and the shock going right through the Spectrum where it blew out every circuit in the machine.



Self-inflicted problems

However, acts of God apart, it does seem that a lot of what goes with Spectrums is, as one company told us, "mostly self-inflicted". There are certain problems, such as keys ceasing to function, that just crop up from time to time and they're plain bad luck, but there is one cardinal sin that you can and should always avoid. **Never plug anything into, or remove anything from, the Spectrum's peripheral port while the power is switched on.**

I know that sounds a bit obvious but everyone we spoke to agreed that 'misuse of the edge connector' is still the number one cause of dead Spectrums. Barry Clayton of Microfare in Bristol told us:

"There are some problems that are very Spectrum specific. They're caused mainly because the Spectrum was built down a low-cost specification. It's often things that could be avoided by having a fuse here or there, as on the C64." As a result, the Spectrum doesn't have some of the safeguards that are built into more expensive computers and other types of electrical appliance, and there are no 'buffers' protecting the chips, which are the heart of any computer, from the raw electrical supply. On the other hand, as Barry Clayton added, "because it's a low-cost machine it's that much easier to repair, so it's a case of swings and roundabouts."

Another similar problem that can be avoided is when the alignment key on an interface falls out, causing a bad connection when the interface is plugged in (if you look at the Spectrum's edge connector you'll see a little notch cut out of the left-hand side, this matches up with a little bar on any interface that is designed for use with the machine. This is the alignment key, a fairly cheap way of ensuring that the proper electrical connection is made when using any peripherals).

Having a go

Then there is what's known as "amateurs having a go." The Quantum repair service told us that quite a lot of things go wrong the moment people start to take the cover off the machine. "Don't touch it — a computer in complexity is 5/6 times more complex than a car engine and there's very few people who could take one apart."

Specialist REPAIR Guide

Contact Nicola Baty on 01-437 0699

QUANTUM ELECTRONIC SERVICES

SINCLAIR SERVICE CENTRE

We have the spares for repairs to:

SPECTRUM/+/+2, QL, MICRODRIVE, INTERFACE 1.

- Fast repair service
- Free estimates
- While you wait service available in most cases. Please phone first.

OR IF YOU WANT TO DO IT YOURSELF!

4164 Memory	£1.90
4116 Memory	£0.99
Z80 A CPU	£2.95

All prices include VAT. Please add £1.00 p&p.

QUANTUM

33 City Arcade, Coventry, CV1 3HX.
Phone: Coventry (0203) 24632

COMPUTER REPAIRS

Spectrum, Commodore, Amstrad, Acorn, BBC etc.
Fixed charge repairs on all makes
Please ring for details

MCE SERVICES

33 Albert Street, Mansfield, Notts NG18 1EA

TEL: 0623 653512

FIXED PRICE SPECTRUM REPAIRS

£19.95 all inclusive

Bring this ad for **£1.00** discount
Open 6 days a week
Do not post — please phone us

0332 365280

FIRST BYTE COMPUTERS
10 Main Centre, London Road, Derby DE1 2PE.

SPECTRUM REPAIRS

£14.95 inclusive of labour, parts and p&p.
Fast, reliable service by qualified engs., average repair **24 hrs. 3 month guarantee.**
For help or advice ring:

H. S. COMPUTER SERVICES,
Unit 2, The Orchard, Warton, Preston, Lancashire PR4 1BE. Tel: (0772) 632686

WHY NOT ADVERTISE IN OUR NEWEST SECTION + ADVERTISE YOUR REPAIR SERVICE?

01-437 0699

COMPUTER RELIABILITY CENTRE

Dear Computer Owner,

Given the choice would you prefer to have your computer repaired inside an hour, and have it sent back to you the same day, probably running the risk of it breaking down again with the same or similar fault within 24 hrs of receipt,

or would you prefer to wait a little longer and have your computer go through our unique validation and rigorous environmental heat soak tests, which highlights any hidden weaknesses within your computer, therefore, increasing the computers **RELIABILITY AND LIFE EXPECTANCY?**

The decision is yours! We are sure that it will be a **RELIABLE** one.

Spectrum 48K all faults	£19.95
16K to 48K upgrade	£24.95
48K to 48K+	£29.95
Spectrum +2	£22.00 min
Amstrad	£22.00 min
Commodore	£24.00 min

Send computer only + cheque to:

BYTES N' BYTES

Block 14, Holden Way Donibristle Ind Est., Dunfermline, Fife, Scotland KY11 5JW.
Telephone: 0383 824576

Note all computers carry a six month warranty

COMPUTER SPARES

Stockists of full range of Sinclair parts.
Trade Counter Open
Mon-Sat 9.30am.-5pm.

P.V. TUBES

104 Abbey Street, Accrington, Lancs BB5 1EE.
Tel: 0254 390936
Telex: 635562 Griffin G (For P.V.)

SPECTRUM REPAIRS

From **£3-£19** incl. all parts, labour and 3 month Guarantee.
Phone for special postal rates

MICRO FARE

296 Gloucester Road, Horfield, Bristol
Tel: 0272 46575

COMPUTER REPAIRS

We are the experts, having serviced Sinclair computers since the introduction of the ZX80.
Don't waste money on estimates — we repair Sinclair computers at price quoted (inclusive parts, labour, postage, VAT, irrespective of fault. No hidden charges).

Repairs guaranteed for 3 months.

Spectrum	£18.75 inc parts
ZX81	£11.50 inc parts
16 KRam	£9.95 inc parts
Microdrive	£15.95 inc parts
Interface 1-11	£18.75 inc parts
also	
BBC	£22.00 + parts
Electron	£19.95 + parts
XK Memory Expansion Kit	£15.95

Computer Retailers please phone for Special Trade Price.

Call or send with cheque or P.O.

T.V. SERVICES OF CAMBRIDGE LTD.
French's Road, Cambridge, CB4 3NP.
Tel: 0223 311371

MICROPOWER

NORTHWOOD HOUSE
NORTH STREET
LEEDS LS7 2AA

Authorised Servicing & Repair Centre

TEL: (0532) 458800

A MSTRAD / CORN / TARI

SPEAK/EASY COMPUTER SERVICES

ALL REPAIRS TO SPECTRUM COMPUTERS

- ★ 3 month full waranty
- ★ Fast reliable repairs by fully qualified engineers
- ★ Open 6 days a week
- ★ Late nights Tuesday & Fridays up to 8pm
- ★ Trade repair enquiries welcome

141 Merry Street, Motherwell, Strathclyde ML1 1JP.
Tel: 0698 53609

NOBLES COMPUTER REPAIRS

- ★ All repairs carry a 4 month guarantee.
- ★ Free estimates.
- ★ 7 days a week, 24 hour answering service.

NOBLES

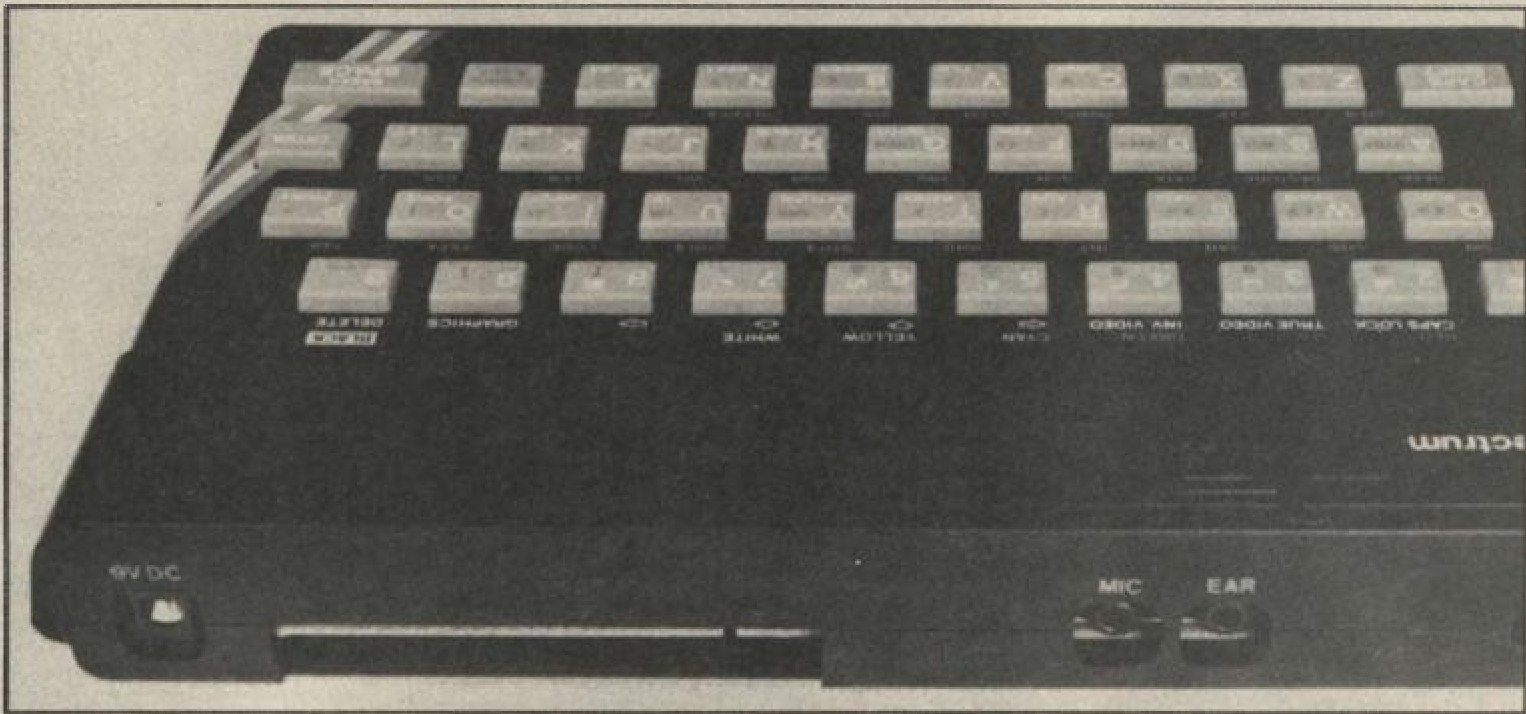
14-17 Eastern Esplanade, Southend-on-Sea, Essex.

Tel: (0702) 63377/8

Keyboard problems crop up from time to time, and the quality of the keyboard was always one of the things that Sinclair machines were criticsed for in the past. Instead of using 'proper' typewriter style keyboards, Sinclair opted for a cheaper, pressure sensitive membrane. The trouble with that was that if you constantly chose a particular key that got used more than the others (if, for example, you always chose the same key as the 'fire' key in arcade shoot 'em ups) then you could literally wear the membrane out. This led to the large market in add-on keyboards. When people getting their machines repaired decided that for a few pounds extra they might as well go the whole hog and buy themselves an entirely new board.

That about covers the problems to which the Spectrum is especially prone. Peripherals like the microdrive and Interface 1 are a different kettle of fish altogether, although some of you might be surprised to hear that repair services don't seem to have too much trouble with these, the general view being that they're OK "if treated properly."

So, despite the poor reputation that Sinclair machines had for reliability in their early days, the repair companies seem to agree with Sir Clive's own claims that much of the problem was due to the wear and tear inflicted on the machines by their owners ("butchered" was the word one company used), rather than poor quality control. The Spectrum may not have that many safeguards built into it, and any mass produced item will throw up faulty units from time to time, but as long as you don't blow it up yourself your Spectrum should stand you in good stead for years to come.



The edge connector: the most vulnerable part of your Spectrum

UK Repair Services

AB COMPUTERS AND ELECTRONICS	01-568-7149
ACE SURE	0613-398266
BULLRING COMPUTERS	0746-266839
G. BUNCE AND SON	0628-661696
CAPITAL COMPUTER SERVICES	0222-461801
CHILTERN COMPUTER CENTRE	0582-455684
COMPUTER HOSPITAL	0964-43354
DIMENSION COMPUTERS	0533-57479
FIRST BYTE	0332-365280
4MAT COMPUTING	0772-561952
GATWICK COMPUTERS	0293-26240
GAZTEK COMPUTERS	0379-52327
H.S. COMPUTER SERVICES	0772-632686
I.T. WESTERN ELECTRONICS	0225-705017
MANCOMP LTD.	0612-241888
M.C.E. SERVICES	0623-31202
MICROCITY	0203-382049
MICROFARE	0272-46575
MICROMEND	0733-241718
MICRO POWER LTD.	0532-458800
MICROSEN	0236-737110
MICRO SERV	0236-737110
MICROTRIX	0204-398176
MICROWORKSHOP	0612-054974
NORTH DEVON MICRO CENTRE	0271-44260
VIC ODENS	01-403-1988
VIDEO VAULT	061-236-0376
ONE STEP BEYOND	0603-663796
PEAK	0429-233199
PROMPT	01-836-7166
QUANTUM	0203-24632
R.A. ELECTRONICS	0502-66289
SCREEN SCENE	0242-528979
SPEAKEASY COMPUTERS	0698-53609
SPECTRUM HOSPITAL	0602-751153
TECHNIFORM SERVICES	0934-843460
THETFORD MICROS	0276-66266
TV SERVICES OF CAMBRIDGE	0223-311371
ZEDEM COMPUTERS	0234-213645

HARDWARE & PERIPHERALS

Where to find the right add-ons and accessories.

ADAPT ELECTRONICS 01-504-2840
Ordinary 48K versions of the Spectrum don't have an RGB output that allows you to plug it into a monitor, so if you want to use one you'll need Adapt's RGB interface.

ADVANCED MEMORY SYSTEMS 0925-413502
Makers of the AMX Mouse.

CHEETAH 0222-777337
Joysticks, interfaces, connectors, you name it, Cheetah make it. But they are best known for their range of musical add-ons such as the SpecDrum, Sound Sampler and their new MK5 Midi Keyboard.

CLASSIFIED PRODUCTS 0325-313131
Suppliers of all sorts of cables and extension leads.

DEAN ELECTRONICS 0344-885661
Probably the only suppliers of thermal paper for the Aphacom 32 printer.

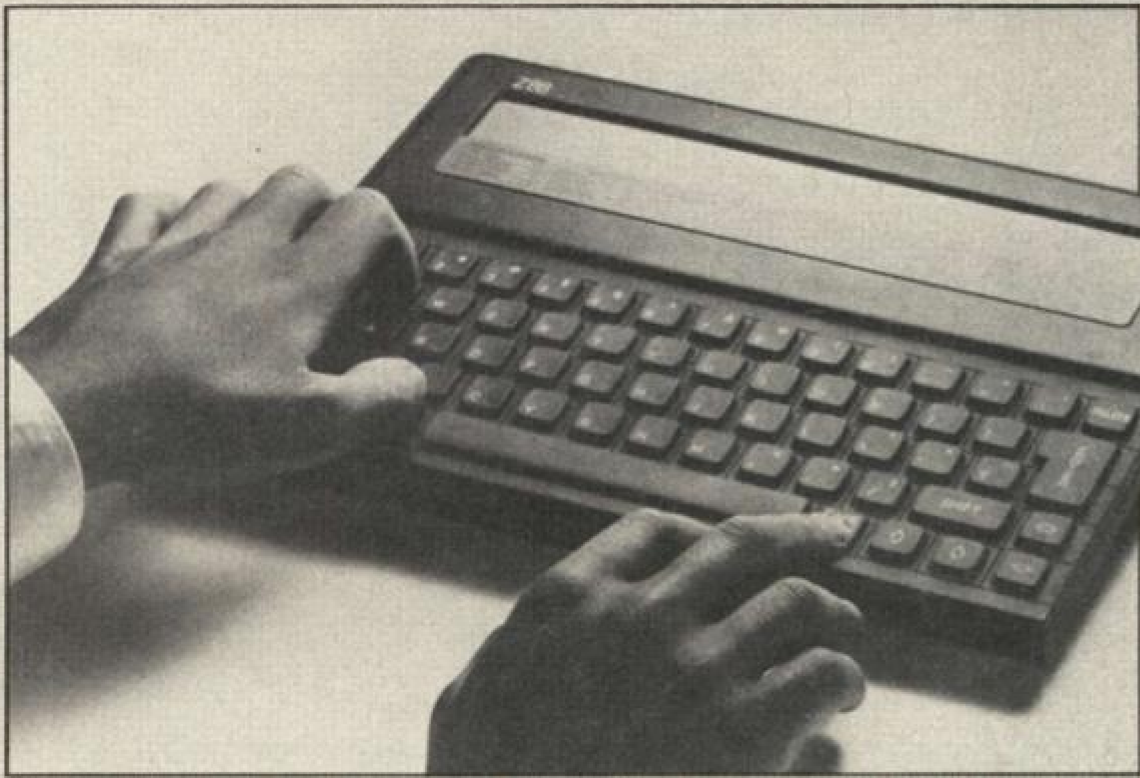
KEMPSTON DATA 0908-690018
As well as producing just about every sort of interface that your Spectrum is ever likely to need, they also do a neat little mouse.

RAM ELECTRONICS 0252-850031
Ram started out by making joystick interfaces but they've recently branched out with the impressive RamWrite printer interface (with built-in word-processor) and the Music Machine Midi interface that doubles as a sound sampler and drum synth.

ROCKFORD PRODUCTS 01-203-0191
Newcomers to the Spectrum market with their Disciple unit which is a combination disc, joystick and networking interface.

ROMANTIC ROBOT 01-200-8870
Producers of the Multiface One unit and now the Mutiface 128.

SAGA SYSTEMS 04862-22978
Have the add-on keyboard market cornered with a whole range of keyboards at different prices. They also do a nice line in printers and wordprocessing equipment.



Sir Clive's new 288 portable

TECHNOLOGY RESEARCH 01-816-3547

TRANSFORM 01-658-6350
Transform don't actually produce anything themselves, but they are one of the best suppliers of printers, monitors, utilities and such like, and they are also one of the few places that can still supply microdrive cartridges.

TROJAN 0792-205491
Makers of one of the better Spectrum lightpens.

VIDEO VAULT 061-236-0376
General repairs company, who also supply the Sinclair's Spectrum+ upgrade kit and the old DK'tronics keyboard — neither of which are generally available these days.

Where are they now?

AMSTRAD CONSUMER ELECTRONICS 0277-228888
They changed the face of the Spectrum with the 128+2, will they change direction with the +3?

CAMBRIDGE COMPUTERS 0223-312216
Followers of Sir Clive will undoubtedly want this number now that he's re-emerged with the new 288 portable which will initially be available by mail order only.



The Spectrum 128+2

SOFTWARE

A directory of British software house contact numbers.



ACTIVISION 01-431-1101
The UK offshoot of a large American outfit, so most of their stuff is conversions of C64 games such as Ballblazer, Hacker and their all-time biggy, Ghostbusters.

ADDICTIVE 0202-296404

ALLIGATA 0742-755796

ADVANCE SOFTWARE 0279-412441
New software house made impressive debut with Hardball

ARGUS PRESS SOFTWARE 0742-755796
Software conglomerate that makes a habit out of swallowing up smaller companies. Quicksilva, Lothlorien, Bug Byte and A 'n F are all members of the APS stable.

ARIOLASOFT 01-836-3411

BUBBLEBUS 0732-355962

CCS 01-858-0763
Strategy/Wargames aren't particularly big business in the UK market, but CCS have managed to specialise in them without going bust. Their latest release, Vulcan is reviewed elsewhere this issue.







CODEMASTERS 0295-68426
Promising new budget label formed by ex Mastertronic programmers.

CRL 01-985-2391
Emerged last year as purveyor of good quality adventures such as the Boggit and Dracula.

CSD 0252-522200

DOMARK 01-947-5626
Masters of the licencing deal, Domark have been responsible for such software classics as Friday The 13th, View To A Kill and the Christmas success, Trivial Pursuit.

DURELL 0823-54489

THE EDGE 01-831-1801

ELECTRIC DREAMS 0703-229694
Another outfit that's fond of licencing deals, Electric Dreams tend to concentrate on film tie-ins (Back To The Future, Howard The Duck, etc), though only the recent Aliens really did justice to the original film.

ELITE 0922-55852

FTL 021-5572981
The arcade label launched by Gargoyle Games who had previously concentrated on adventure games. There's only been one FTL release so far, the cracking Lightforce, but there's more to come.

GARGOYLE GAMES 0215-572981

GILSOFT 0446-732765
Gilsoft's entire reputation and business is based on the success of their adventure creating system, The Quill, which has spawned a whole mini-industry of budget adventures. The Quill's getting a bit old now, so watch out for the arrival of the new Professional Writing System.

GREMLIN 0742-753423
The software house that Monty Mole built

HEWSON 0235-832939
Andrew Hewson's outfit has built up a solid reputation based on good games. The latest is Steve Turner's Runarama.

HI SOFT 0525-718181
Latest product is the ZX Basic Compiler.

INCENTIVE 0734-591678
Makers of the Graphic Adventure Creator.

LEVEL 9 0734-595759
With no competition in sight, Level 9 are our top adventure company. Their Middle Earth and Silicon Dream trilogies are landmarks in the field. These days they're spreading their wings by doing link-ups with other companies, such as Knight Orc, due from Rainbird in April.



MASTERTRONIC 01-377-6880
The UK's top budget software house, so successful in fact that they've just gone and bought out the once proud ...

MELBOURNE HOUSE 01-943-3911

MIKROGEN 0344-427317

MICROSPHERE 01-883-9411

MIRRORSOFT 01-377-4837

OCEAN 0619-803488
High output of coin op conversions and spin-offs. Notable games recently, Cobra and Great Escape.

PALACE 01-278-0751
A subsidiary of the Palace Films/Video outfit, Palace have tended to stick to a small number of good releases such as the Cauldron games and the recent Antiriad, though they may be expanding in the future.

PIRANHA 01-836-6633
An offshoot of the MacMillans publishing company. Best game to date is the quirky Trapdoor.

PSS 020-3667556

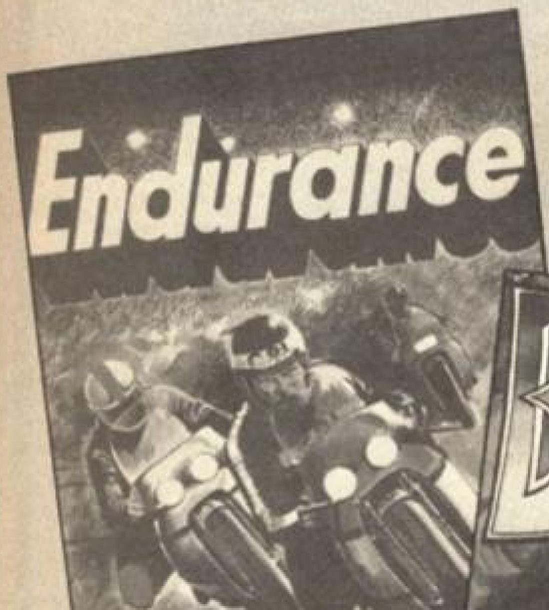
SOFTWARE PROJECTS 051-4289393
Since making Matthew Smith's fortune with Manic Miner and Jet Set Willy, they've had what is kindly referred to as a 'low profile,' though their Dragon Lair conversions show that there's life in them yet.

TASMAN 0532-438301
Makers of multifarious Tas — utilities such as Tas Copy, Tas Print, but most notably Tasword now in its third incarnation.

TELECOMSOFT 01-240-8838
Telecomsoft is the little bit of British Telecom that owns or markets **Odin, Beyond, Firebird** and **Rainbird** (who are conveniently profiled in this issue).

U.S. GOLD 021-3563388
Close runners to Ocean in the big league, U.S. Gold have got a variety of labels under their wing, including marketing deals with Vortex, Ultimate and Adventuresoft, as well as a seemingly endless store of U.S. software waiting to be converted onto the Spectrum.

VIRGIN 01-834-8341
The software arm of the Branson empire. Last year snapped up Leisure Genius and thereby the rights to titles such as Scrabble, Monopoly and Scalextric.





Brian Becket looks at the latest QL software.

QL COLUMN

ZX Microfairs continue to showcase new products although it seems as if there are less visitors each time. The only advantage to come out of this is the relative ease of walking around to see what's on offer without all the pushing and shoving through hordes of games nuts. There wasn't much new in the way of the QL either at the most recent Microfair, but any news is good news these days and it is impressive how new products keeping coming despite Amstrad having officially sealed the coffin.

Digital Precision worked all night and had the long-awaited **Turbo-charge,** son-of-Supercharge mega-compiler out on the day. Although my review copy arrived a day or so after the show, I am sadly unable to say much more about Turbo than I said last month as unfortunately I was sent the wrong size disc.

I did, however, get Digital's final benchmarks for Turbo and can pass them on as hot-off-the-presses news. According to Digital, average speed-up factors for compiled code are: (1) Integer benchmarks x84 for Turbo compared to x41 for Supercharge, (2) floating point benchmarks x11.7 for Turbo compared to x7.2 for Supercharge. Digital's estimates for its competitor QLiberator are x11.2 and x4.1 respectively. For a "Superbasic long program (Project Planner)", Digital's estimates are QLiberator x7, Supercharge x12 and Turbo-charge x45. For comparing compiler speeds, Digital takes QLiberator as x1 and puts Supercharge at x1.7 and Turbo at x2.3. Digital was over the moon about the results and — up to the last minute — the programmers and designers didn't know how good a product they had developed. They expected Turbo to be as fast or faster then Supercharge but confessed to pure delight when the results came in. Turbo looks an excellent piece of software and the company deserves every success with it.

Making headlines

The Front Page is not just the title of a classic play and a good movie but also a package of QL software offering desk-top publishing for a mere £22.50. Produced by GAP Software, the Front Page is a no-frills program that is fully menu-driven and enables the user to produce text and graphic layouts freely and easily. It comes on one microdrive and works with an unexpanded QL but is designed to take advantage of disc, RAM discs, and so on if you have the facilities. The Front Page is designed to offer a reasonably powerful desk-top publishing at a bargain basement price to those who want to produce their own news letters, handouts or whatever.

GAP's MD had read my QL obituary a few months back and told me that — while he would defend to the death my right to say it — I ought to be taken out and shot. He had noted a growing QL enthusiasm among church groups, school groups and such like and I pointed out that this could be due to the fact that the QL had reached rock-bottom prices in the last few months. This combined with some of the truly excellent software available (including the Front Page) makes the QL an attractive proposition to low-budget groups and others outside the IBM life-style. At the end of the day, he forgave me (I think) for being a journalist — especially after I said that I was fully behind all small QL companies and users — and we parted as friends. But the QL is dead and — unless somebody pulls off a minor miracle with Amstrad, it's going to stay that way. This doesn't mean that the machine won't be useful to a lot of people for a long time or that it's pointless to produce any more new QL products but only that it's out of the race in the marketplace.

Speaking of new products, the keyboard stakes are getting hotter. There was one selling for around £50 at the Microfair. It just gives the old keys a better feel and frankly I don't think it's worth £50 so I don't plan to say any more about it. Schön has won the race to get their first and its replacement boards are now available. They sell for just over £40 and you get a new QL housing with a new set of LED wires plus the typewriter-like keyboard. You just take off the old QL top, slot in the new LED wires (the keyboard comes with a clear instruction sheet) and the new keyboard tails. After replacing the screws, you're ready to go. It's an impressive product and a good bargain. There's no soldering to worry about — the only tool you need is a small Phillips-head screwdriver — and the result gives you a QL with a keyboard of professional quality. The main drawback I found with Sir Clive's keyboard was the tendency to hit two keys at the same time but Schön's replacement avoids this and you can type with an ease that's a pleasure by comparison. The board's layout is exactly the same (the function keys are in red but the rest is in the black that we all know and love) and the board is compatible with all peripherals and software.

QL casino

If you fancy running a small casino for your family, neighbours and anybody with a few quid to lose, Pyramid has just released **Super Croupier** at £15. You play against the QL but there's nothing stopping you from letting the computer be the House backed up by your cash but this would be illegal and — as QL owners are an honest lot — the point is of academic interest only. There are six games on the program: roulette, blackjack, baccarat, carta + alta, poker and slot machine. Personally, I like the Poker and the Baccarat because I've never had any luck with Blackjack or Roulette in real casinos. Baccarat is a very enjoyable game (it can also be a costly one) and it's nice to have a computer program to play against. It's a very good package (you even get Snoopy running around the screen to help out) and well worth the money.

line

A programming utility from Kenneth Baker.

A good Line Renumber program is, without doubt, one of the most useful programs that anyone could wish to add to their collection. However, the problem with most utilities is that they demand a fixed position in memory, and usually require a small BASIC program to operate them. This is not the case with this program — because it can be loaded to any convenient position in memory at any time: it also contains routines to tell itself where it is located in memory, and to renumber its own variables accordingly. It also has its own input routines, dispensing with the need for a BASIC control program.

The routines in the main part of the program have been designed to cater for every imaginable situation: all forms of GOTO, GOSUB, RUN, RESTORE and SAVE . . . LINE statements can be renumbered (including microdrive), even when in complex conditional form.

Add to this the data for various captions, and the original renumber routine, which is a mere thirty-three bytes in length, grows into a somewhat larger 1140 bytes!

```
   10 REM ***********************
   20 REM **     HEX  LOADER    **
   40 REM ***********************
   50 CLEAR 59999
   60 FOR F=60000 TO 61140 STEP 8
   70 PRINT F;
   80 LET CHECK=0
   90 INPUT "Enter bytes:"; LINE
H$
  100 IF LEN H$<>16 THEN  PRINT "
Error in length": BEEP .2,0: GO
TO 70
  110 PRINT TAB 8;H$;
```

Before we can go any further, however, it will be necessary to enter the HEX LOADER program and the machine code. The LOADER program has been designed to ensure that the process of entering the machine code is error-free, rejecting any lines that are not correctly entered, or in the wrong order.

The HEX CODE is arranged in 127 lines of eleven pairs. The first nine pairs represent the actual machine code instructions, whilst the last two pairs contain a checksum of the values in the particular line. If any other method of input is used you should remember to ignore these last two pairs.

Each line of eleven pairs should be entered as a continuous string, *with one space between each pair.* The lines are listed in convenient blocks to assist with their location during input. The address to be inserted in Line 4 of the hex loader must be the location at which you have decided to enter the machine code, say 60000 for 48K machines. Line 5 sets RAMTOP to one less than this address. When loading has finished the program should be saved on tape by using: **SAVE "renumber" CODE 60000, 1140.**

You can now experiment witt the program by renumbering th HEX LOADER program. Use RANDOMIZE USR 60000 (or whatever address you have chosen) and, if everything is in order, a prompt will appear in the lower part of the screen requesting you to: **ENTER STARTING LINE, STEP SIZE**

This requires you to enter two values separated by a comma, followed by ENTER. The values entered must be such that the STEP SIZE is not greater than the STARTING LINE, although the two can be equal. Otherwise you can enter any values you like — 1,1 — 999,99 — 9000,2 etc. If you type in rubbish, illegal values, or ENTER with no values at all, the system will automatically use the values 10,10.

Directly the renumbering process has been completed (in a fraction of a second with an average program) the program title will appear on the screen, where it will remain until a key is pressed. The only restrictions concerning the use of the program are that the first line of a program to be renumbered must contain a REM statement, and corresponding addresses must exist for any commands involving line numbers: a statement like GOTO 100 would be ignored if Line 100 does not exist.

Normally, assuming that the program is safely stored on cassette, it can be loaded to any position in memory by using the following sequence:

1. Choose a suitable address
2. CLEAR address
3. LOAD "" CODE address
4. Make sure that first line of program is a REM
5. **RANDOMIZE USR address**
6. ENTER STARTING LINE, STEP SIZE

Machine code buffs will detect instantly that the same address has been used for all three commands: this is permissible in this case because the first instruction in the program has the same code as the RAMTOP marker, LD, A,N. RAMTOP can of course be set at any address you desire, provided that it is somewhere below the concerned.

Display file

It was mentioned earlier that this program could be entered anywhere — it can also be stored in the Display File (at address 19000) in cases where renumbering is an afterthought, or where you only intend to use the program once. You will not need to use the CLEAR command with this method, but you must use CLS first, so try:

CLS
LOAD "" CODE 19000
RANDOMIZE USR 19000

and the program will execute in the normal way. You will have to reload the program each time you use this method, as the program will vanish directly you enter a command, but the method does have many advantages — principally that, because CLEAR is not used, all of the variables are left intact, and you are not left with the unwanted renumber program in the computer once it has performed its brief task.

If you do ever wish to remove the code from the computer, the following command is suggested:

FOR N=address TO address+1140: POKE n,0: NEXT n

```
120 FOR G=1 TO 8
130 LET BYTE=16*(CODE H$(1)-48-(7 AND H$(1)>"9"))+(CODE H$(2)-48-(7 AND H$(2)>"9"))
140 POKE F-1+G,BYTE
150 LET CHECK=CHECK+BYTE*G
160 LET H$=H$(3 TO )
170 NEXT G
180 INPUT "Enter check digit:"; LINE C$
190 IF LEN C$<>1 THEN  GO TO 180
200 PRINT TAB 27;" ) ";C$
210 LET DIGIT=CHECK-9*INT (CHECK/9)
220 IF DIGIT<>VAL C$ THEN  PRINT "Error in string - input again": BEEP .2,-12: GO TO 70
230 NEXT F
240 PRINT  FLASH 1;"READY TAPE FOR SAVING"
250 SAVE "renumber"CODE 60000,1140
260 CLS
270 PRINT  FLASH 1;"READY TAPE TO VERIFY"
280 VERIFY "renumber"CODE 60000,1140
```

renumber_

```
60000   3FFF21005B36E123   ) 2
60008   36E52336C9CD005B   ) 7
60016   111000A7ED521854   ) 3
60024   000000006868805A   ) 6
60032   1C024EAA00130000   ) 7
60040   0504801300801000   ) 3
60048   80AD0000C63A80F0   ) 2
60056   0C00420080CC4C01   ) 4
60064   8D00819900001E00   ) 0
60072   802700002A038024   ) 7
60080   0001A00081990000   ) 2
60088   080001196C000600   ) 3
60096   80120000600080E5   ) 6
60104   62810E26E5111800   ) 3
60112   19E5C1E111D50019   ) 6
60120   E5111D00A7ED52EB   ) 3
60128   E13E1CF5D50A5703   ) 3
60136   0A5F03CB7ACBBA28   ) 7
60144   05A7ED52180119D1   ) 3
60152   E50A0326006F19EB   ) 8
60160   E17323722BF13D20   ) 3
60168   DA2A535C06042310   ) 5
60176   FD7EFEFA282A181C   ) 5
60184   4E4F20414354494F   ) 0
60192   4E3A0D4649525354   ) 4
60200   CAC3EA5354415445   ) 3
60208   4D454E542118EB06   ) 6
60216   1C7ED72310FB183A   ) 7
60224   08F5CD9CEBF10821   ) 7
60232   D3ED11C0500602C5   ) 2
60240   D50604C5D506207E   ) 1
60248   12D51412D1132310   ) 2
60256   F6D11414C110ECD1   ) 5
60264   06201310FDC110DF   ) 1
60272   21C05A0640367223   ) 3
60280   10FB010000C33D1F   ) 8
60288   494E505554205354   ) 1
60296   4152542D4C494E45   ) 3
60304   2C20535445502053   ) 3
60312   495A450D2180EB06   ) 6
60320   1C7ED72310FB2100   ) 2
60328   5BE5E50600360023   ) 1
60336   10FBF1CDDE153A08   ) 2
60344   5CFE0D280CFE2C28   ) 5
60352   08FE3038FEFE3A30   ) 5
60360   FAF5E52101011101   ) 7
60368   00CDB503F1F1FE0D   ) 0
60376   2809E5D7E1D63077   ) 3
60384   231AD036FCF17FFF   ) 3
60392   FC28391602010000   ) 4
60400   7EFFFC2815D5F560   ) 3
60408   6950592929192906   ) 0
60416   004F09444DF1D123   ) 7
60424   18F623152806FD43   ) 8
60432   DCFC18D9FD43DFFC   ) 8
60440   2ADCFCFD5BDFFCA7   ) 7
60448   FD52301221DCEC11   ) 7
60456   DFFC3E0A77231213   ) 6
60464   3F00771218F222DC   ) 5
60472   FC2ADFFCAFBF28E4   ) 4
60480   2A535C1A02232323   ) 7
60488   2323237FF5FD5B4B   ) 0
60496   5CA7ED52E1D2ADFD   ) 0
60504   FE0D28EAFE0E28E5   ) 5
60512   FEEC2810FEED280C   ) 0
60520   FEF72808FECA2804   ) 4
60528   FEE520D6237FFE0E   ) 3
60536   280AFE0D28C8FE3A   ) 7
60544   28C818F0232323E5   ) 1
60552   5E2356D5D5E1181E   ) 2
60560   3E04A7110A00ED52   ) 7
60568   3810115A00ED5238   ) 0
60576   0A118403ED523804   ) 6
60584   18033D3D3DC9CD90   ) 1
60592   EC08D12A535C0100   ) 5
60600   00037E23BA20047E   ) 0
60608   BB281823D55E2356   ) 8
60616   2319E5A7ED5B4B5C   ) 4
60624   ED52E1D138E308E1   ) 1
60632   C34AEC210000110A   ) 5
60640   00190B78B120FAAF   ) 2
60648   EBE1732372D5E5EB   ) 3
60656   CD90ECE1D12B2B2B   ) 8
60664   2B47084FB0281AD5   ) 7
60672   C5160059A7ED5242   ) 5
60680   CDE819C1D1C5D548   ) 5
60688   0600CD5516EB23D1   ) 6
60696   C1C5782B10FDE5C1   ) 5
60704   D5E1E5FE04280AFE   ) 3
60712   03280CFE02280E18   ) 1
60720   1211E803CD4BED11   ) 7
60728   6400CD4BED110A00   ) 5
60736   CD4BED110100CD4B   ) 6
60744   ED180EAFED523803   ) 6
60752   3C18F919C6300203   ) 3
60760   C9D1C5E1C1E53E0D   ) 1
60768   2BBE20FC232323D5   ) 0
60776   5E2356E5EB160059   ) 0
60784   A7ED525819EBE172   ) 5
60792   2B73D1E123232323   ) 1
60800   23237EFE2AC24AEC   ) 5
60808   237EFE2B280CFE0D   ) 0
60816   CA46ECFE3ACA4AEC   ) 7
60824   18EE237EFE0ECA84   ) 1
60832   ECFE0DCA46ECFE3A   ) 1
60840   CA4AEC18ED2A535C   ) 2
60848   ED4BDCECED5BDFEC   ) 7
60856   EB09EB424B702371   ) 7
60864   235E23562319ED5B   ) 0
60872   4B5CA7ED52300319   ) 8
60880   18E2C90000000000   ) 8
60888   0000000000000000   ) 0
60896   3C00000000000000   ) 6
60904   0000000000000000   ) 0
60912   000000401000007C   ) 5
60920   0000000020000000   ) 7
60928   42183C3C08004400   ) 6
60936   0000001040007C00   ) 1
60944   2000004000783842   ) 2
60952   3878446020381C00   ) 6
60960   9928424218004838   ) 3
60968   7878383840004238   ) 2
60976   28381C4030444442   ) 2
60984   444444543C442000   ) 2
60992   A108423C28007044   ) 1
61000   4444441078007C04   ) 1
61008   304420401044787C   ) 3
61016   7844445422782000   ) 2
61024   A1083E4240004878   ) 3
61032   444478104400423C   ) 2
61040   3078204010444044   ) 5
61048   4044445422402000   ) 6
61056   990802427E004440   ) 5
61064   4444401044004244   ) 6
61072   2840207E38443C42   ) 1
61080   3C4438543C3C2000   ) 0
61088   423E3C3C0800423C   ) 8
61096   44443C0C44007C3C   ) 5
61104   243C200000000000   ) 0
61112   0000000000000000   ) 0
61120   3C00000000000000   ) 6
61128   0000000000000000   ) 0
61136   000000002A535C01   ) 1
```

technical graphics

Part 3: Toni Baker goes wandering into some 3D landscapes.

Welcome to the final part of this series. Take a look at the screen dump included in this article — this is just one possible output of the program listed here. As you can see, it has a number of interesting properties. Firstly, it is a three-dimensional graph. Specifically it is a graph of the function.

Z= SIN(R)
Where R = SQR(X^2+Y^2)

The function, as well as the boundaries and scales of the graph, can be easily changed. The program is therefore capable of handling almost *any* three-dimentional graph. There are of course restrictions on what is possible. The function must be of the form *Z=f(X,Y)* — that is to say, there must exist an algorithm to convert given values of X and Y to a single value of Z.

But notice something else about the graph. Hidden detail really is hidden. Anything which falls behind a peak or ridge is hidden from view, and is not drawn on the screen. Of course — this isn't a *true* hidden line algorithm. It cannot draw solid figures — it is restricted to drawing one very special kind of three dimensional landscape, a graph of a function of X and Y.

The last subroutine in the machine code listing (the part labelled DEMO_FN at address 827A) is not actually part of the main program at all. It is merely the algorithn for calculating Z, given X and Y, for the function Z= 3 SIN(R)/R (with R = SQR(X^2Y^2)). You can replace, or supplement, this subroutine with similar functions of your own, providing you follow the RULES, which I shall now explain:

```
  10 RANDOMIZE FN G(-10,10,10,10
,1,1,40,33402)
  20 STOP
  30 DEF FN G(X,X,X,X,X,X,X,X)=U
SR 33380
```

Figure 1.

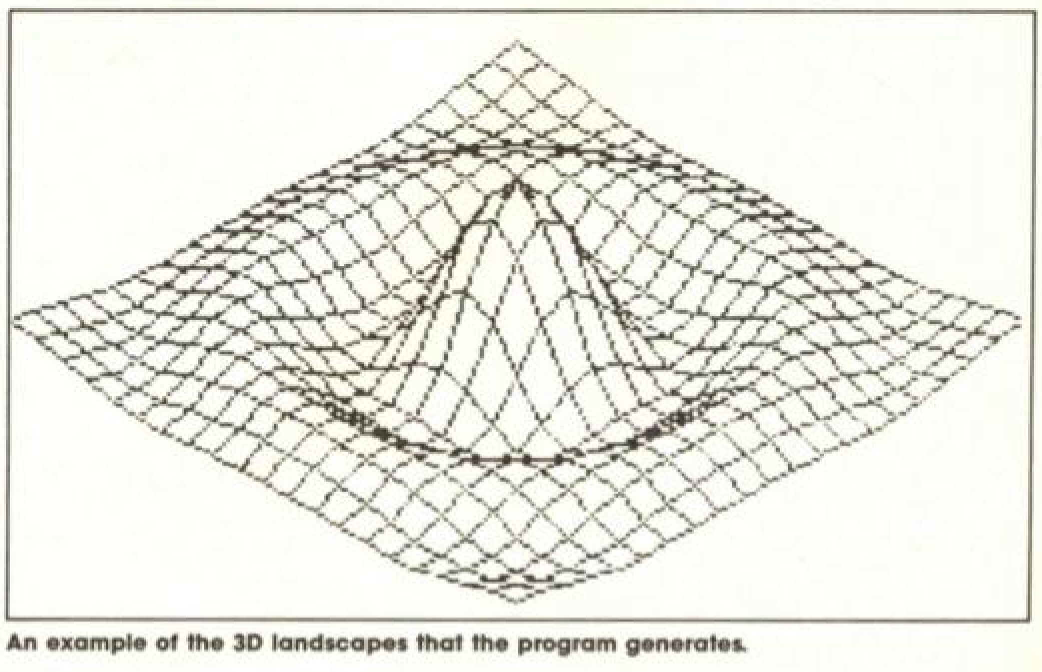
Rule one — the subroutine must assume that on entry the calculator stack is empty, containing no entries whatsoever.

Rule two — you are allowed to make use of calculator memories M0 to M3, but memories M4 upwards are strictly out of bounds, and must not be corrupted.

Rule three — sixteen memories altogether are used by the main program, numbered M0 to MF. You will need to read at least two of them — memory ME contains the variable X, and memory MF contains the variable Y. Your algorithm must read these memories whenever X and Y are required.

Such a subroutine may be placed at any address, and therefore it is possible to contain a library of such functions in memory simultaneously.

Once such a function subroutine exists in memory, it is then easy to draw the graph of it. Figure One shows the BASIC program required to draw the screen image shown. More generally, Figure Two shows the BASIC line required to draw any graph; it is a use of the BASIC function FN G, requiring eight parameters. The last of these parameters must be the address of the machine code subroutine for calculating the Z values, Figure Two explains the purpose of each of the eight parameters, with reference to Figure Three,



An example of the 3D landscapes that the program generates.

Figure 2.

```
DEF FN G(X,X,X,X,X,X,X,X) = USR 33380

RANDOMIZE FN G(X1,Y1,X2,Y2,XH,YH,Zscale,FNaddr)
```

FNaddr = address of machine code subroutine to calculate Z values of function.

Zscale = factor by which Z coordinates are to be multiplied.

(X1,Y1,X2,Y2,XH,YH) See Figure Three.

Listing

```
                        ORG 8100
EF       G_PLOT         RST 28                  Switch calculator on.
E4                      recall M4               plotX
E5                      recall M5               plotX,plotY
38                      endcalc                 Switch calculator off.
CDD52D                  CALL 2DD5,FP_TO_A       A:= plot Y coordinate.
F5                      PUSH AF                 Stack this value.
CDD52D                  CALL 2DD5,FP_TO_A       A:= plot X coordinate.
3819                    JR C,G_PL_EXIT          Jump if X coord too big.
2017                    JR NZ,G_PL_EXIT         Jump if X coord negative.
6F                      LD L,A                  L:= plot X coordinate.
2680                    LD H,80                 HL: points to entry in column table.
F1                      POP AF                  A:= plot Y coordinate.
C0                      RET NZ                  Return if Y coord negative.
3804                    JR C,G_PL_MAX           Jump if Y coord much too big.
FED0                    CP D0
3802                    JR C,G_PL_OK            Jump unless Y coord too big.
3EFF     G_PL_MAX       LD A,FF                 A:= FF to signal "above screen".
BE       G_PL_OK        CP (HL)
D8                      RET C                   Return if point to be plotted
                                                is "hidden".
77                      LD (HL),A               Store new value for this column.
47                      LD B,A
4D                      LD C,L                  C,B:= plot coordinates.
3C                      INC A
C4E522                  CALL NZ,22E5,PLOT_SUB   Plot point unless above screen.
C9                      RET                     Return.
F1       G_PL_EXIT      POP AF                  Balance the stack.
C9                      RET                     Return.

                        ORG 8128
EF       G_DRAWTO       RST 28                  to-X,to-Y
E5                      recall M5               to-X,to-Y,plotY
C3                      store M3                (M3 stores lineY, =plotY).
03                      subtract                to-X,drawY
C1                      store M1                (M1 stores drawY, = Y disp for DRAW).
01                      exchange                drawY,to-X
E4                      recall M4               drawY,to-X,plotX
C2                      store M2                (M2 stores lineX, =plotX).
03                      subtract                drawY,drawX
C0                      store M0                (M0 stores drawX, = X disp for DRAW).
07                      or                      drawX<>0 OR drawY<>0 ?
0004                    jump true,G_DR_CONT_1   Jump unless both displacements zero.
38                      endcalc
C9                      RET                     Return.
EF                      RST 28
E1       G_DR_CONT_1    recall M1               drawY
2A                      abs                     ABS(drawY)
E0                      recall M0               ABS(drawY),drawX
2A                      abs                     ABS(drawY),ABS(drawX)
03                      subtract                ABS(drawY)-ABS(drawX)
36                      lt zero                 ABS(drawY)< ABS(drawX)?
                                                =HV, say.
31                      duplicate               HV,HV
000D                    jump true,G_DR_CONT_2   Jump if line horizontal-ish.
E0                      recall M0               HV,drawX
E1                      recall M1               HV,drawX,drawY
C0                      store M0                (M0:= number of points to plot).
                                                (-ve if line goes
02                      delete                  HV,drawX
C1                      store M1                (M1: contains transverse disp).
02                      delete                  HV
E2                      recall M2               HV,lineX
E3                      recall M3               HV,lineX,lineY
C2                      store M2                (M2:= forward coord of last plot).
02                      delete                  HV,lineX
C3                      store M3                (M3:= transverse coord of last plot).
02                      delete                  HV
E1       G_DR_CONT_2    recall M1               HV,dispT
E0                      recall M0               HV,dispT,dispF
05                      divide                  HV,dispT/dispF
C1                      store M1                (M1:= ratio).
02                      delete                  HV
E0                      recall M0               HV,dispF
38                      endcalc                 Switch calculator off.
00                      NOP
CDD52D                  CALL 2DD5,FP_TO_A       A:= number of points to plot.
F5       G_DR_LOOP      PUSH AF                 Stack number and direction.
2009                    JR NZ,G_DR_NEG          Jump if line to be drawn backwards.
EF                      RST 28                  Switch calculator on.
E2                      recall M2               HV,lineF
A1                      const one               HV,lineF,1
0F                      add                     HV,lineF+1
E3                      recall M3               HV,lineF+1,lineT
E1                      recall M1               HV,lineF+1,lineT,ratio
0F                      add                     HV,lineF+1,lineT+ratio
3308                    jump G_DR_CONT_3        (Jump forward).
EF       G_DR_NEG       RST 28                  Switch calculator on.
E2                      recall M2               HV,lineF
A1                      const one               HV,lineF,1
03                      subtract                HV,lineF-1
E3                      recall M3               HV,lineF-1,lineT
E1                      recall M1               HV,lineF-1,lineT,ratio
03                      subtract                HV,lineF-1,lineT-ratio
C3       G_DR_CONT_3    store M3                (M3:= new transverse plot position).
C5                      store M5                (M5:= new transverse plot position).
02                      delete                  HV,lineF±1
C2                      store M2                (M2:= new forward plot position).
C4                      store M4                (M4:= new forward plot position).
```

SPECTRUM PROGRAMMING

```
02                      delete                  HV
31                      duplicate               HV,HV
0007                    jump true,G_DR_CONT_4   (Jump if line horizontal-ish).
E4                      recall M4               HV,plotX
E5                      recall M5               HV,plotX,plotY
C4                      store M4                (M4:= new plot X coordinate).
02                      delete                  HV,plotX
C5                      store M5                (M5:= new plot Y coordinate).
02                      delete                  HV
38        G_DR_CONT_4   endcalc                 switch calculator off.
CD0081                  CALL 8100,G_PLOT        Plot next point on line.
C1                      POP BC
05                      DEC B                   B:= no of remaining points to plot.
C5                      PUSH BC
2803                    JR Z,G_DR_EXIT          Jump if all points plotted.
F1                      POP AF                  Zero flag:= direction of line.
18D2                    JR G_DR_LOOP            Loop back to draw all points.
EF        G_DR_EXIT     RST 28
02                      delete
38                      endcalc
F1                      POP AF                  Balance the stack.
C9                      RET                     Return.
                        ORG 818B
210080    G_CLEAR       LD HL,8000              HL: pts to 1st byte of column table.
110180                  LD DE,8001              DE: pts to 2nd byte of column table.
01FF00                  LD BC,00FF              BC:= length of col table, less one.
70                      LD (HL),B               Reset first byte.
EDB0                    LDIR                    Reset remaining bytes.
C9                      RET                     Return.

                        ORG 8198
2A925C    G_CONVERT     LD HL,(MEMBOT)          HL:= address of function subroutine.
CD2C16                  CALL 162C,CALL_JUMP     Call this subroutine.
EF                      RST 28                  Z
EC                      recall MC               Z,Zscale
04                      multiply                Z'   (=Z*Zscale).
E8                      recall M8               Z',X2
E9                      recall M9               Z',X2,Y2
0F                      add                     Z',X2+Y2
EE                      recall ME               Z',X2+Y2,X
EF                      recall MF               Z',X2+Y2,X,Y
0F                      add                     Z',X2+Y2,X+Y
03                      subtract                Z',(X2+Y2)-(X+Y)
ED                      recall MD               Z',(X2+Y2)-(X+Y),XYscale
04                      multiply                Z',((X2+Y2)-(X+Y))*XYscale
0F                      add                     Z'+((X2+Y2)-(X+Y))*XYscale
A2                      const half              Z'+((X2+Y2)-(X+Y))*XYscale,1/2
0F                      add                     Z'+((X2+Y2)-(X+Y))*XYscale+1/2
27                      int                     plotY
E8                      recall M8               plotY,X2
E7                      recall M7               plotY,X2,Y1
03                      subtract                plotY,X2-Y1
EE                      recall ME               plotY,X2-Y1,X
EF                      recall MF               plotY,X2-Y1,X,Y
03                      subtract                plotY,X2-Y1,X-Y
03                      subtract                plotY,(X2-Y1)-(X-Y)
34F15DB3
D745                    stk data SQR(3)         plotY,(X2-Y1)-(X-Y),SQR(3)
                         (41 5D B3 D7 45)
04                      multiply                plotY,((X2-Y1)-(X-Y))*SQR(3)
ED                      recall MD               plotY,((X2-Y1)-(X-Y))*SQR(3),XYscale
04                      multiply                plotY,((X2-Y1)-(X-Y))*SQR(3)*XYscale
A2                      const half              plotY,((X2-Y1)-(X-Y))*SQR(3)*XYscale
                                                ,1/2
0F                      add                     plotY,((X2-Y1)-(X-Y))*SQR(3)*XYscale
                                                +1/2
27                      int                     plotY,plotX
01                      exchange                plotX,plotY
38                      endcalc                 Switch calculator off.
C9                      RET                     Return.

                        ORG 81C4
CD9881    G_PLOT_3D     CALL 8198,G_CONVERT     Calculate PLOT coordinates.
EF                      RST 28                  plotX,plotY
C5                      store M5                (M5: contains plotY).
02                      delete                  plotX
C4                      store M4                (M4: contains plotX).
02                      delete
38                      endcalc                 Switch calculator off.
C30081                  JP 8100,G_PLOT          Jump to plot point.

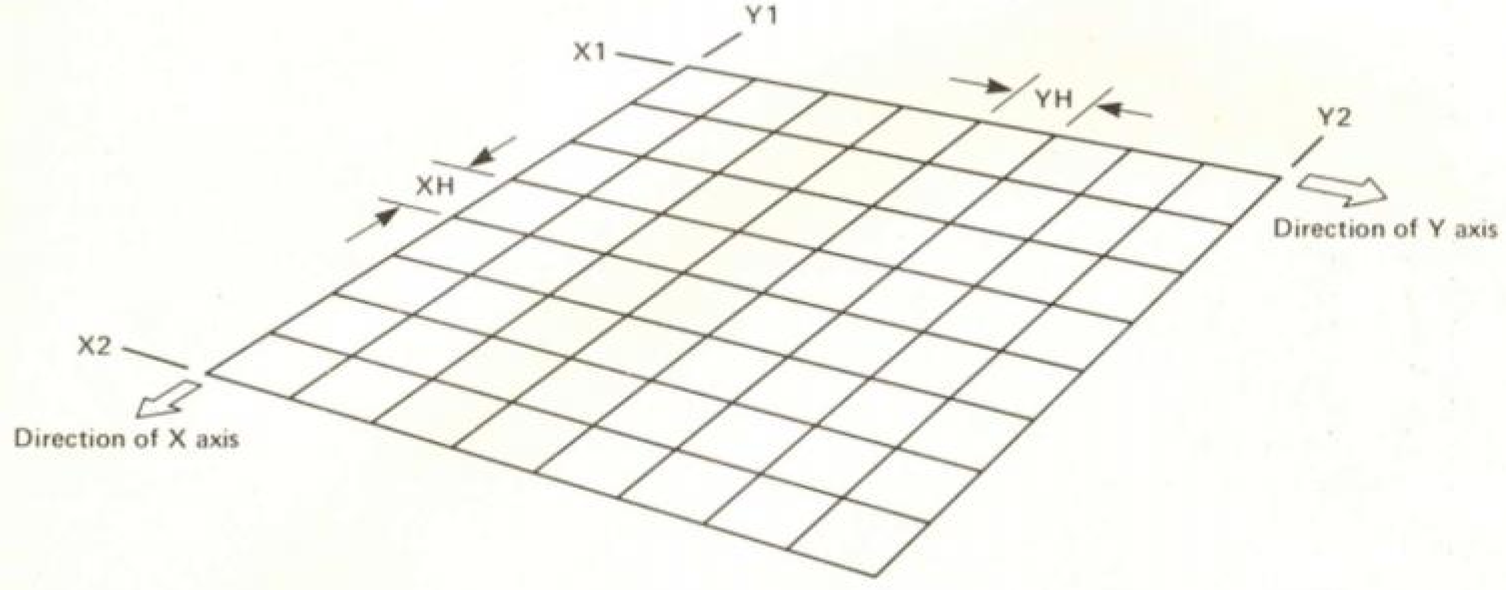
                        ORG 81D0
CD9881    G_DRAWTO_3D   CALL 8198,G_CONVERT     Calculate DRAW_TO coordinates.
CD2881                  CALL 8128,G_DRAWTO      Draw the line.
CD541F                  CALL 1F54,BREAK_KEY     Is BREAK key pressed?
D27B1B                  JP NC,1B7B,REPORT_L     Generate BREAK report if so.
C9                      RET                     Otherwise return.
                        ORG 81DD
E5        G_GRID        PUSH HL                 Stack subroutine address.
CD8B81                  CALL 818B,G_CLEAR       Clear column table.
015000                  LD BC,0050              BC:= no of bytes reqd for memories.
F7                      RST 30                  Create room for memories.
EB                      EX DE,HL                HL: points to memory zero.
22685C                  LD (MEM),HL             Store in system variable.
0E41                    LD C,41                 BC:= 0041.
09                      ADD HL,BC               HL: points to memory MD.
EB                      EX DE,HL                DE: points to memory MD.
0E24                    LD C,24
2A655C                  LD HL,(STKEND)          HL: points beyond calculator stack.
EDB8                    LDDR                    Copy top seven items from calculator
                                                stack into memories M6 to MC.
23                      INC HL
22655C                  LD (STKEND),HL          Remove these items from calc stack.
E1                      POP HL                  HL:= subroutine address.
22925C                  LD (MEMBOT),HL          Store in (MEMBOT) temporarily.
EF                      RST 28                  Switch on the calculator.
34F813CD
3A2C                    stk data 256/SQR(3)     256/SQR(3)
                         (88 13 CD 3A 2C)
E8                      recall M8               256/SQR(3),X2
E6                      recall M6               256/SQR(3),X2,X1
03                      subtract                256/SQR(3),X2-X1
E9                      recall M9               256/SQR(3),X2-X1,Y2
E7                      recall M7               256/SQR(3),X2-X1,Y2,Y1
03                      subtract                256/SQR(3),X2-X1,Y2-Y1
0F                      add                     256/SQR(3),(X2-X1)+(Y2-Y1)
05                      divide                  256/(SQR(3)*((X2-X1)+(Y2-Y1)))
CD                      store MD                (Memory MD contains XYSCALE).
02                      delete
E9                      recall M9               Y2
CF                      store MF                (Memory MF contains Y, =Y2).
02                      delete
E8        G_LOOP_Y1     recall M8               X2
CE                      store ME                (Memory ME contains X, =X2).
02                      delete
38                      endcalc                 Switch calculator off.
CDC481                  CALL 81C4,G_PLOT_3D     Plot first point on curve.
EF                      RST 28                  Switch calculator on.
EE        G_LOOP_X1     recall ME               X
EA                      recall MA               X,XH
03                      subtract                X-XH
CE                      store ME                (Decrement X value by XH).
E6                      recall M6               X,X1
03                      subtract                X-X1
36                      lt zero                 X<X1?
0008                    jump true,G_EXIT_1      Jump if X now less than X1.
38                      endcalc                 Switch calculator off.
CDD081                  CALL 81D0,G_DRAWTO_3D   Draw next segment of curve.
EF                      RST 28                  Switch calculator on.
33F1                    jump G_LOOP_X1          Jump back to continue drawing curve.
EF        G_EXIT_1      recall MF               Y
EB                      recall MB               Y,YH
03                      subtract                Y-YH
CF                      store MF                (Decrement Y value by YH).
E7                      recall M7               Y,Y1
03                      subtract                Y-Y1
36                      lt zero                 Y<Y1?
30                      not                     Y>=Y1?
00DF                    jump true G_LOOP_Y1     (Loop back if Y still in range).
38                      endcalc                 Switch calculator off.
CD8B81                  CALL 818B,G_CLEAR       Clear column table again.
EF                      RST 28                  Switch calculator on.
E8                      recall M8               X2
CE                      store ME                (Memory ME contains X, =X2).
02                      delete
E9        G_LOOP_X2     recall M9               Y2
CF                      store MF                (Memory MF contains Y, =Y2).
02                      delete
38                      endcalc                 Switch calculator off.
CDC481                  CALL 81C4,G_PLOT_3D     Plot first point on curve.
EF                      RST 28                  Switch calculator on.
EF        G_LOOP_Y2     recall MF               Y
EB                      recall MB               Y,YH
03                      subtract                Y-YH
CF                      store MF                (Decrement Y value by YH).
E7                      recall M7               Y,Y1
03                      subtract                Y-Y1
36                      lt zero                 Y<Y1?
0008                    jump true,G_EXIT_2      (Jump if Y now less than Y1).
38                      endcalc                 Switch calculator off.
CDD081                  CALL 81D0,G_DRAWTO_3D   Draw next segment of curve.
EF                      RST 28                  Switch calculator on.
33F1                    jump G_LOOP_Y2          Jump back to continue drawing curve.
EE        G_EXIT_2      recall ME               X
EA                      recall MA               X,XH
03                      subtract                X-XH
CE                      store ME                (Decrement X value by XH).
E6                      recall M6               X,X1
```

```
03                    subtract               X-X1
36                    lt zero                X<X1?
30                    not                    X>=X1?
00DF                  jump true,G_LOOP_X2    (Loop back if X still in range
                                             to draw next X curve).
38                    endcalc                Switch calculator off.
21925C                LD HL,MEMBOT
22685C                LD (MEM),HL            Restore calculator memories.
C9                    RET                    Return.

                      ORG 8264
2A0B5C    FN_GRID     LD HL,(DEFADD)         HL: pts to user defined FN records.
23        FN_G_LOOP   INC HL
23                    INC HL                 HL: points to next argument.
CDB433                CALL 33B4,STACK_NUM    Push argument onto calculator stack.
7E                    LD A,(HL)
23                    INC HL                 HL: points to next record.
FE2C                  CP 2C
28F5                  JR Z,FN_G_LOOP         Loop back until all args on calc stk.
CD991E                CALL 1E99,FIND_INT2    BC:= function subroutine address.
60                    LD H,B
69                    LD L,C                 HL:= function subroutine address.
C3D081                JP 81D0,G_GRID         Jump to draw grid.
                      ORG 827A

EF        DEMO_FN     RST 28                 Switch calculator on.
E0                    recall MX              X
31                    duplicate              X,X
04                    multiply               X^2
E1                    recall MY              X^2,Y
31                    duplicate              X^2,Y,Y
04                    multiply               X^2,Y^2
0F                    add                    X^2+Y^2
28                    sqr                    R        (=SQR(X^2+Y^2)).
31                    duplicate              R,R
30                    eq zero                R,R=0?
0008                  jump true,DF_ZERO      (Jump if R=0).
31                    duplicate              R,R
1F                    sin                    R,SIN(R)
01                    exchange               SIN(R),R
05                    divide                 SIN(R)/R
38                    endcalc                Switch calculator off.
C9                    RET                    Return.
EF                    RST 28
02        DF_ZERO     delete
A1                    const one              1
                                             (=limit of SIN(R)/R as R→0).
38                    endcalc                Switch calculator off.
C9                    RET                    Return.
```

Figure 3.



which explains six of them diagramatically.

Figure Three in fact shows a bare grid, without a function imposed on it. (You could argue of course that it shows a graph of the function Z = 0, but that would be trivial). This grid is converted into a 3D graph (such as the one whose screen dump is shown) by elevating (or perhaps lowering) each point on the grid by the Z value calculated for that point. The grid will therefore "warp" into a three dimensional graph.

Although the scale for the Z axis must be included by hand as one of the parameters for FN G, the scales for the X and Y axes are worked out automatically by the program. The scales are chosen so that the graph fits exactly across the full width of the screen ensuring maximum possible resolution.

All you actually have to worry about is the range of values allowed by X, and the range of values allowed by Y. In this program the X values range from X1 to X2, in steps of XH, while the Y values range from Y1 to Y2 in steps of YH, so you don't necessarily have to start at the origin (although of course you can if you want to).

Algorithms

Let's have a look at the main program itself, and see how its algorithm works. The first thing to note is that the graph is drawn in two passes. In pass one the lines parallel to the X axis are drawn, and in pass two the lines parallel to the Y axis are drawn. In each case the lines (or to be more precise, curves) are drawn from front to back. It is *because* the curves are drawn in this order that the elimination of hidden detail is possible.

If any part of a line falls behind already existing detail, then the hidden part must not be drawn. By working from front to back this decision is easy to make. If a line is intended to be drawn *above* all other parallel lines, then the line will be visible, and must be drawn. On the other hand, if a line is intended to be drawn *below* any of the parallel lines, then it is hidden and must not be drawn.

In practice, however, whole lines are not always either completely above, or completely below, other detail. In practice

technical graphics

only *part* of a line must be drawn. this means that the DRAW algorithm in the ROM is no good to us, and we must rewrite one of our own. The decision process of whether an individual dot composing part of a line is to be plotted or not is still, however, tremendously easy.

What we do is to construct a table, at address 8000h, of 100h bytes. Each entry in the table corresponds to a pixel-column from the screen. Whenever we plot, or attempt to plot, a point in the screen, then we must examine the table entry corresponding to the column number of the point. The table contains the height, in pixels, of the uppermost plotted point on the screen in the given column. If the point to be plotted lies above this point then it may be safely plotted and the table entry updated. If the point to be plotted lies below this point then it is hidden, and should not be plotted. If we now take a look at the machine code itself, we shall see this algorithm in detail. I shall explain the workings of each subroutine as we come to it.

Routines

The first subroutine is called G_PLOT (address 8100). This is the subroutine which attempts to PLOT a point (whose screen coordinates are held in memories M4 and M5). It compares the coordinates given with the column table, in the manner described above, and PLOTs the point only if it is not hidden.

The second subroutine is G_DRAWTO (address 8128). This is my own version of the DRAW algorithm. It will draw a line beginning at the point whose screen coordinates are held in memories M4 and M5, and ending at the point whose screen coordinates are at the top of the calculator stack, in the order X,Y. Instead of actually plotting each point on the line, however, it will subject each point to the G_PLOT algorithm above, to decide whether or not it is hidden.

Next we have a nice easy subroutine, G_CLEAR (address 818B). All this does is to empty the column table ready for each pass.

The next subroutine is called G_CONVERT (address 8189). Its purpose is to convert three-dimensional graph coordinates to two-dimentional screen coordinates. It takes the graph coordinates X and Y from memories ME and MF respectively, calculates the graph coordinate Z using the subroutine provided by the user, and will then proceed to calculate the screen coordinates *plotX* and *plotY* from the graph coordinates X, Y and Z, using an isometric projection algorithm similar to that in Part Two of this series. On exit, the screen coordinates *plotX, plotY*, are left on the calculator stack, in that order.

Then we have the subroutine G_PLOT_3D (address 81C4), which is used to plot the endpoints of the lines. On entry the graph coordinates X,Y must be stored in memories ME and MF. The subroutine will convert these to screen coordinates before attempting to plot the point.

G_DRAWTO_3D (address 81DO) is very similar. It takes the graph coordinates X and Y from memories ME and MF, and converts them to screen coordinates, and then proceeds to draw a line from sceen coordinates M4,M5 to the point calculated, hiding any points which need to be hidden. The subroutine also tests whether or not BREAK is pressed, so that you can break out the program before it's finished by pressing BREAK or CAPS-SHIFT/SPACE.

The subroutine G_GRID (address 81DD) is the main algorithm for drawing the grid. It is in three parts. The first part is concerned with initialisation. It creates the sixteen memories required by the program and initialises memories M6 to MD. The values for M6 to MC are taken from the calculator stack, and are the first seven parameters supplied by FN G. The value for MD has to be calculated, and this is done here. The final two parts are the first and second pass of grid drawing. On each pass an outer loop varies Y (or X) for each curve, whilst an inner loop varies X (or Y) for different point along that curve. Finally, the calculator memories are restored to normal. Note that on entry HL must contain the function subroutine address.

The final subroutine in the program is called FN_GRID (address 8264), and it is this subroutine which transfers the parameters supplies by FN G onto the calculator stack (and the last one into HL) before leaping into G_GRID to draw the actual graph.

The subroutine DEMO_FN (address 827A7 is not a part of the main program. It is, rather, an example program, intended to be replaced by your own efforts, as described earlier.

This program concludes the TECHNICAL GRAPHICS series. We shall give graphics a break, for a while, but graphics enthusiasts have no fear. A new series entitled 3D-GRAPHICS is planned for the not-too-distant future. Until then, there'll be some surprises. Good programming everyone.

Figure Four

These are the calculator memories used by the program. Note that the first six of these variables are multi-purpose, so that different variables are stored there at different times.

Any line-segment drawn by the program is defined as being either "horizontal-ish" or "vertical-ish", depending on its slope. for "horizontal-ish" lines, the FORWARD coordinate is always the X coordinate, and the TRANSVERSE coordinate is always the Y coordinate. For "vertical-ish" lines the reverse is true, with the FORWARD coordinate being the Y coordinate, and the TRANSVERSE coordinate being the X coordinate.

M0	drawX	DRAW displacement in X direction.
	dispF	DRAW displacement in forward direction.
M1	drawY	DRAW displacement in Y direction.
	dispT	DRAW displacement in transverse direction.
	ratio	The ration dispT/dispF.
M2	lineX	X coordinate of next point on line.
	lineF	Forward coordinate of nex point on line.
M3	lineY	Y coordinate of next point on line.
	lineT	Transverse coordinate of next point on line.
M4	plotX	X coordinate of point to PLOT.
	plotF	Forward coordinate of point to PLOT.
M5	plotY	Y coordinate of point to PLOT.
	plotT	Transverse coordinate of point to PLOT.
M6	X1	Lower bound for X (see Figure 3).
M7	Y1	Lower bound for Y (see Figure 3).
M8	X2	Upper bound for X (see Figure 3).
M9	Y2	Upper bound for Y (see Figure 3).
MA	HX	Step size dividing X axis (see Figure 3).
MB	YH	Step size dividing Y axis (see Figure 3).
MC	Zscale	Amount by which Z coords must be multiplied.
MD	XYscale	Half of amount by which X and Y coordinates must be multiplied.
ME	X	General X coordinate (3D).
MF	Y	General Y coordinate (3D).

Lettahead+/Dumpy
Bradway Software
£8.50/£9.00

Recently, Carol Brooksbank gave some beautiful examples of screen dumps, posters and tickets, with hints on how to produce them (ZX, Feb 87). One of the difficulties of this sort of thing is that you need a good collection of dump routines to fit your printer/interface combination. As there are lots of interfaces and printers, you can be left with quite a problem to get your masterpiece printed, sometimes soluble, it's true, but only if you are prepared to spend a lot of time.

Bradway Software have been producing programs that are, literally, just the ticket. Two programs, in particular, sort out virtually all these sorts of problems, enabling you to product a variety of machine code dumps in an enormous range of printer/interface combinations.

Letta-head plus is designed primarily to allow you to print letterheads. labels or small tickets. A special feature lets you produce labels, three in a row if necessary, and a variant of this gives you ready-outlined cassette labels.

Cassette, microdrive or Opus disc versions of the programs are available for an additional £1.50. Typical of the care taken to cover all combinations is the inclusion of a Wafadrive version on side B of the cassette; this is because the Wafadrive uses much more of the Spectrum RAM for its system variables, buffers and directories than a microdrive, and it was therefore impossible to include the Wafadrive printer interface version within the main program.

The program itself comes with a choice of five assorted print styles already available, and a further twenty replacements which can be selected from a menu and loaded: or you can load in your own favourite fonts or use a font-editing/designing program (also supplied) to create your own designs. The program enables you to print to any part of the screen in one of these fonts, put a box round it, invert an area, scroll any part of the screen in any direction (grand if you want to centre things), invert or erase an area and put in a grid to help planning. The art subroutines are pretty rudimentary (plot, draw, arc, circle and rapid fill); this is not a substitute for a graphics program, nor was it intended to be so. Any SCREEN$ from such a program can, however, be loaded, edited and used as part of the display: a scratch-pad/memory is provided to help you with this and allow you to merge one SCREEN$ with another.

A strength and a weakness of the program is that a large amount is in BASIC, which means that it is very user friendly (although system errors have not been trapped), readily adapted, and, most important of all, can be restarted on crashing without losing the picture. It also means that cursor movement, erasing and printing to the screen are all fairly slow, and one would not want to use this program for printing out many pages of fancy text.

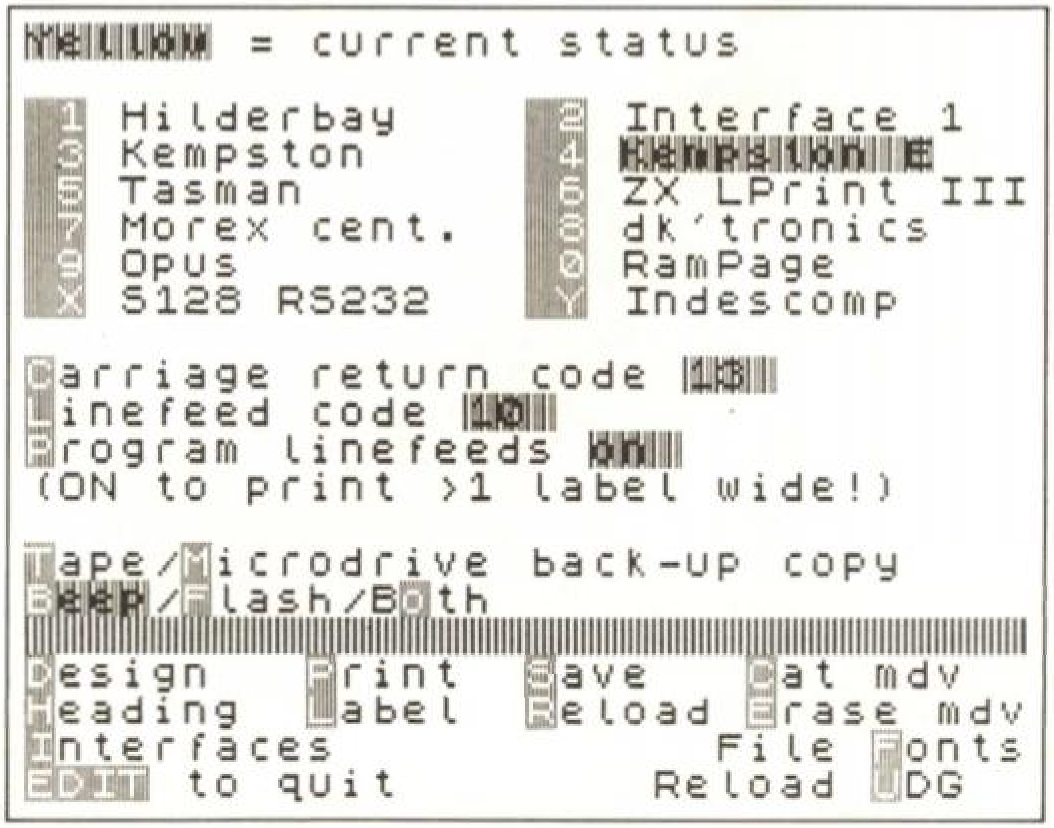
The best feature is the way the program will deal with a huge range of printer/interface combinations. In addition there are hints and tips for dealing with "not quite Epson-standard" printers including the Brother and Centronics GLP, and even the "not nearly Epson-standard" Smith Corona Fastext 80, for which a special machine code patch is included. All in all, I found this to be a utility program which has had a great deal of thought put into it, is very well documented and, within its limitations, does what it was intended to do with a minimum of user-hassle, and I would recommend it.

Unlike lettahead, Dumpy does not have built-in type fonts and design capabilities. What it does is to assemble from its comprehensive library of short routines a range of stand-alone, relocatable machine-code screen-dump programs which will suit your printer/interface combination. Dumpy allows you to choose between normal or sideways dumps, plain or shaded dumps, and there is a whole range of widths, heights and printer densities to choose from. In addition, one can specify an area of screen and use an auto-start to skip over blank lines at the start or finish of the picture.

If you have done something silly like trying to locate the machine code in the middle of the BASIC program area, a large black warning appears on screen, but you are still allowed to proceed: in other words it is designed both for the nincompoop and those who blind us with science. Finally, you get the option of saving the code and/or testing it, for which a coloured screen is loaded.

Again, this program has been carefully put together with a comprehensive 12-page handbook neatly printed in stiff card covers and including a helpline for those in need.

Both programs are available from Bradway Software on cassette at £8.50 for Dumpy 3 and £9.00 for lettahead+, microdrive cartridges or Opus discs £1.50 extra, including post and packing.



STREAMS AND CHANNELS

This month, Toni Baker opens up a new channel to the 128's RAMdisc facility.

This is the final part in the Streams and Channels series, and this is an article for people who own either a Spectrum 128 or a Spectrum 128 +2. This month's new channel introduces SERIAL FILES to the 128K machines.

People who have Interface Ones and Microdrives will already be used to serial files on microdrive cartridge. In this case you open a microdrive file either as a READ file (if the file already exists on microdrive) or a WRITE file (in which case the file must be created on microdrive). You can print text to a WRITE file, and then CLOSE and OPEN it (so that it becomes a READ file) and you may then input your text as either strings or numbers into a BASIC variable.

RAMdisc

RAMdisc files work in exactly the same way, except that you don't need an Interface One. When you first OPEN a RAMdisc file for writing, a file with a given name is created on the Spectrum's so-called 'silicon disc'. Text or numbers may then be printed into this file. Once the channel is closed no more printing to the file is possible. The file may, however, be re-OPENed as a read-file, in which case whatever is in the file may be input into a BASIC variable.

As with microdrive files, RAMdisc files must be CLOSEd once all the data has been printed to, or input from, the file. If a WRITE file is not closed then some or all of the data may be lost, as it will not be cleared from a special buffer. If a READ file is not closed then the consequences are less serious, however, each RAMdisc channel requires more than ½K, which may only be reclaimed by closing the file. You should always close such a file once you have finished with it.

Once a file has been opened, it will appear in the RAMdisc catalogue, which you can verify by typing CAT! in BASIC. It is impossible to LOAD a RAMdisc serial file using a LOAD command, however it *is* possible to ERASE a RAMdisc serial file in the usual manner, by entering ERASE! "filename". You should never ERASE a RAMdisc serial file which is still in use (ie. which still has a stream attached to it). The machine code program does in fact protect itself from this eventually, so such an error would not be fatal, however — you will certainly get spurious results if you break this rule.

Silicon disc

The key to how the machine code program works is the manipulation of the memory organisation known as the silicon disc, or RAMdisc, which is normally used to save programs, data, or machine code for as long as the machine is switched on. RAMdisc files are much faster than microdrive files, but the whole of RAMdisc is erased when the machine is switched off. RAMdisc serial files will, of course, suffer from precisely these advantages and disadvantages.

The RAMdisc memory itself is primarily organised by the CATALOGUE, which is an index to all files saved in RAMdisc. The CATALOGUE resides in RAM page seven. It is effectively a stack, which begins at address 7EBFF and grows downwards, with each entry taking twenty bytes. Figure Two shows the meanings of these twenty bytes, with IX pointing to the first of these bytes. At the end of the catalogue stack is a twenty byte "End-of-catalogue" marker, only three bytes of which are used. The system variable (SFNEXT) points to this marker, and is effectively the stack-pointer for this catalogue stack.

So long as we keep the catalogue in its required format, we can manipulate the RAMdisc organisation itself from machine code. This is the aim of this issue's program.

The RAMdisc files themselves begin in RAM page 1, and grow upwards through RAM pages 3, 4, 6 and 7 (care is taken to ensure that RAMdisc files do not collide with the catalogue stack). To avoid any problems with this strange page-numbering, all addresses in RAMdisc are "page-coded". This means that one register will hold a page-code, while another register-pair will hold an actual physical address. These page codes are 0, 1, 2, 3 and 4 sequentially for RAMdisc files, with page-code 5 being the conventional notation for normal (48K) RAM.

Channel 'R'

We shall call our new channel "R", which stands for RAMdisc serial-file. This is not to be confused with the ROM's internal "R" channel, which is used for inserting bytes into normal (48K) dynamic RAM.

R CHANNEL INFORMATION BLOCK

```
IX + 00:     R_OUT          Address of RAMdisc file output routine (=B92B).
IX + 02:     R_IN           Address of RAMdisc file input routine (=B7D4).
IX + 04:     R_NAME         Name of channel (= "R").
IX + 05:     R_IDEN         New channel identifier (= 1234h).
IX + 07:     R_CLOSE        Address of RAMdisc file close routine (= B960).
IX + 09:     R_LEN          Length of channel information block (= 021B).
IX + 0B:     R_CHBYTE       Pointer into buffer.
IX + 0D:     R_CHREC        Record number within file.
IX + 0E:     R_CHNAME       Filename.
IX + 18:     R_CHFLAG       Various flags, defined as follows:
    Bits 7 to 2:      Not used.
    Bit 1:            Set if End-Of-File at end of record, reset otherwise.
    Bit 0:            Reset for a READ file; set for a WRITE file.
IX + 19:     R_RECLEN       Length of record within buffer.
IX + 1B:     R_BUFFER       Buffer storing current record.
```

Figure 1

Internal-channel "R" is quite interesting in fact. It is permanently attached to stream minus-once, so to use it from machine code all you have to do is select stream minus-one as the current stream (by loading A with FF and calling address 1601h). It is impossible to use from BASIC. Prior to selecting stream minus-one the system variable K_CUR must be made to point somewhere into dynamic RAM. Printing to stream FF will then insert characters into dynamic memory at the point indicated by (K_CUR).

Our new channel is also called "R", but its use is much more exciting — and of course it MAY be used from BASIC.

We require a channel information block over ½K in size. Most of this is in fact a huge 0200h byte buffer. For reasons of speed, the buffer is used most of the time, with RAMdisc itself only being accessed once the end of the buffer is reached. It is of course important to realise that the location of a RAMdisc file is not constant — it may move either if another file is erased, or if more bytes are inserted into another RAMdisc serial file. This means that the file has to be relocated every time we wish to read or write into it. To save time we make use of a temporary buffer virtually all of the time.

Figure One shows the actual organisation of the channel information block for our R channel. Note that R_CHREC, R_RECLEN, and bit one of R_CHFLAG are used only if the file is a READ file. The remaining variables are used for both types of file.

To interface with BASIC, an example machine code program is appended to the end. Essentially, a RAMdisc serial file may be opened by loading the A register with the stream number of the stream to be opened, whilst the actual filename is stored in the system variable N_STR1 at address 5B67, and then calling the label R_OPEN. Four additional entry points are included, labelled OPEN_4, OPEN_5, CLOSE_4 and CLOSE_5. Calling OPEN_4 will open a RAMdisc serial file called "FILE1" and attach it to stream four. Similarly, calling OPEN_5 will open a RAMdisc serial file called "FILE2" and attach it to stream five. The routines CLOSE_4 and CLOSE_5 will of course close these new channels.

This means that the new channel may easily be used in BASIC. Take a look at Figure Three. It contains a BASIC program which demonstrates the RAMdisc serial files at work, first as WRITE files, then as READ files. Try it — you may be surprised at how fast it all works.

Of course you won't always want your files to be called "FILE1" or "FILE2", and you won't always want to use streams four or five. That is why the more general entry point R_OPEN is included, which, as has already been stated, requires that A

CATALOGUE INDEX INFORMATION

```
IX + 00:     SF_NAME        Filename.
IX + 0A:     SF_START       Page-coded address of start of file.
IX + 0D:     SF_LEN         Total number of bytes in file, including header info.
IX + 10:     SF_END         Page-coded address of byte beyond end of file.
IX + 13:     SF_FLAG        Reset unless catalogue information incomplete,
                            (ie reset normally).
```

Figure 2

STREAMS AND CHANNELS

contains the stream number, and (N__STR1) contains the filename (with trailing spaces if required). You will have to write your own machine code to patch any other combination into BASIC, utilising this routine.

The potential for streams and channels is limitless. It is theoretically possible, for instance, to have a channel which utilises a RANDOM ACCESS FILE, or INDEXED FILE, in RAMdisc — though the program would have to be much more complicated. I have shown you enough of the potential use for streams and channels to whet your appetite a little, and there I shall leave you. If demand is high enough, I may return with more. Good programming everyone, and may the force be with you.

```
1000 REM WRITE FILE DEMO
1010 RANDOMIZE USR 47713: REM OPEN #4,"R_FILE1"
1020 RANDOMIZE USR 47720: REM OPEN #5,"R_FILE2"
1030 FOR I = 1 TO 512
1040 INPUT "": PRINT I
1050 PRINT #4;2*I
1060 PRINT #5;I*I
1070 NEXT I
1080 RANDOMIZE USR 47736: REM CLOSE #4
1090 RANDOMIZE USR 47740: REM CLOSE #5
1100 REM READ FILE DEMO
1110 RANDOMIZE USR 47713: REM OPEN #4,"R_FILE1"
1120 RANDOMIZE USR 47720: REM OPEN #5,"R_FILE2"
1130 FOR I = 1 TO 512
1140 INPUT #4;A
1150 INPUT #5;B
1160 PRINT A,B
1170 NEXT I
1180 RANDOMIZE USR 47736: REM CLOSE #4
1190 RANDOMIZE USR 47740: REM CLOSE #5
1200 STOP
```

Figure 3

ERRATUM

In OPEN_NEW at address B06D (Streams and Channels Part Two, January Issue, page 68) there is an instruction missing. The instruction is POP BC (hex code C1). It should be the eleventh instruction of the routine, occurring between OR C and JR Z,OPEN_NEW_2. Sorry about that.

Figure 4

The following are vectored routines in the Spectrum's ROM 0. Notice that there are two alternative such vector tables, one for the Spectrum 128 and one for the 128+2. This is because the two machines have different ROMs. If the ROM of the 128+2 is changed at some indefinate point in the future, then it will be necessary to rewrite this table further.

```
                    ORG B6FC (Spectrum 128 only)
C3AC05   V_ERROR    JP 05AC           Generate an error report.
C3641C   V_PAGE     JP 1C64           Change current RAM page.
C3971C   V_NEWCAT   JP 1C97           Create new entry in catalogue.
C3F31C   V_SPACE    JP 1CF3           Ensure enough space in RAMdisc area.
C3121D   V_FIND     JP 1D12           Find catalogue entry for filename.
C3561D   V_CATEND   JP 1D56           Tidy up last catalogue entry.

                    ORG B6FC (Spectrum 128 + 2 only)
C3CB05   V_ERROR    JP 05CB           Generate an error report.
C3831C   V_PAGE     JP 1C83           Change current RAM page.
C3B61C   V_NEWCAT   JP 1CB6           Create new entry in catalogue.
C3121D   V_SPACE    JP 1D12           Ensure enough space in RAMdisc area.
C3311D   V_FIND     JP 1D31           Find catalogue entry for filename.
C3751D   V_CATEND   JP 1D75           Tidy up last catalogue entry.
```

The rest of the program will be the same, whichever version of the Spectrum you have. The following subroutine will decrement a page-coded address held in register-triplet BHL.

```
[illegible]         ORG B70E
2B       DEC_BHL    DEC HL            Decrement HL.
78                  LD A,B            A:= page code.
FE05                CP 05
C8                  RET Z             Return if using standard RAM.
CB74                BIT 6,H
C0                  RET NZ            Return unless HL has crossed
                                      a page boundry.
CBF4                SET 6,H           Correct address in HL.
05                  DEC B             Decrement page code.
C9                  RET               Return.
```

The following subroutine works a bit like a glorified LDDR instruction, which works in RAMdisc area as well as in standard memory. Its action is threefold: (1) Decrement BHL and BHL'; (2) Load one byte from address (BHL) to address (BHL'); (3) If BHL is not equal to CDE then go to step (1).

```
                    ORG B71A
CD0EB7   R_TRANSFER CALL B70E,DEC_BHL Decrement page-coded address in BHL.
D9                  EXX
CD0EB7              CALL B70E,DEC_BHL Decrement page-coded address in BHL'.
D9                  EXX
78                  LD A,B            A:= paging code of FROM address.
CDFFB6              CALL B6FF,V_PAGE  Page in the FROM memory page.
7E                  LD A,(HL)         A:= byte to load.
F5                  PUSH AF           Stack this byte.
D9                  EXX
78                  LD A,B            A:= paging code of TO address.
CDFFB6              CALL B6FF,V_PAGE  Page in the TO memory page.
F1                  POP AF            A:= byte to load.
77                  LD (HL),A         Load byte into memory as required.
D9                  EXX
```

```
78         R_TRANSFER_2   LD A,B                 A:= paging code of FROM address.
B9                        CP C
20E6                      JR NZ,R_TRANSFER       Loop back if not equal to the
                                                 paging code of the limiting address.
ED52                      SBC HL,DE              Set zero flag if address = limit.
19                        ADD HL,DE              (ADD HL,DE does not affect zero flag)
20E1                      JR NZ,R_TRANSFER       Loop back unless limit has been
                                                 reached.
C9                        RET                    Return.
```

The next subroutine will calculate the page-coded address which is BC bytes further on from AHL. It assumes that BC is always less than 4000h.

```
                          ORG B73A
09         ADD_AHL,BC     ADD HL,BC              Increment HL by BC bytes.
FE05                      CP 05
C8                        RET Z                  Return if using standard RAM.
CB74                      BIT 6,H
C0                        RET NZ                 Return unless page boundry crossed.
CBFC                      SET 7,H
CBB4                      RES 6,H                Correct address in HL.
3C                        INC A                  Increment page code.
C9                        RET                    Return.
```

This subroutine will search for the file whose name is specified in the channel information area, in the RAMdisc catalogue, giving an error if the file does not exist. On exit IX will point to the catalogue entry.

```
                          ORG B747
DDE5       FIND_FILE      PUSH IX
E1                        POP HL                 HL: points to channel information.
010E00                    LD BC,000E
09                        ADD HL,BC              HL: points to filename.
0E0A                      LD C,0A                BC:= length of filename (ten).
11675B                    LD DE,5B67,N_STR1      DE: points to system variable.
EDB0                      LDIR                   Copy filename into system variable.
CD08B7                    CALL B708,V_FIND       Find catalogue entry for this name.
C0                        RET NZ                 Return if file exists, with IX
                                                 pointing to catalogue entry.
CDFCB6                    CALL B6FC,V_ERROR      Generate error message
23                        DEFB 23                "h File does not exist".
```

The next subroutine is designed to match up the buffer for R-channel with the corresponding region of RAMdisc memory. The subroutine will leave BHL pointing to the first byte beyond the RAMdisc segment, CDE pointing to the start of the RAMdisc segment, and BHL' pointing to the first byte beyond the corresponding region in the R-channel buffer. It will also signal whether or not this is an end-of-file block. It requires that IX initially points to the channel information block.

```
                          ORG B75D
DDE5       R_MATCH        PUSH IX                Stack channel info address.
DD7E0D                    LD A,(R_CHREC)
F5                        PUSH AF                Stack record number to match.
CD47B7                    CALL B747,FIND_FILE    IX: points to catalogue entry.
C1                        POP BC                 B:= record number.
CB20                      SLA B                  B:= record number x2.
0E01                      LD C,01                BC:= 200h * record number + 1.
37                        SCF
08                        EX AF,AF'              Signal "End of file block".
DD6E0D                    LD L,(SF_LEN)
DD660E                    LD H,(SF_LEN+1)
DD7E0F                    LD A,(SF_LEN+1)        AHL:= length of file (17 bit).
A7                        AND A
ED42                      SBC HL,BC
DE00                      SBC A,00               AHL:= length of remainder of file.
A7                        AND A
2008                      JR NZ,R_M_NO_EOF       Jump if high part of AHL is non-zero.
110102                    LD DE,0201
ED52                      SBC HL,DE
19                        ADD HL,DE
3804                      JR C,R_M_EOF           Jump if AHL less than 0201h.
210002     R_M_NO_EOF     LD HL,0200             HL:= length of record (0200h max).
08                        EX AF,AF'              Signal "Not end of file block".
EB         R_M_EOF        EX DE,HL               DE:= length of record.
DD6E0A                    LD L,(SF_START)
DD660B                    LD H,(SF_START+1)
DD7E0C                    LD A,(SF_START+2)      AHL:= coded address of start of file.
CD3AB7                    CALL B73A,ADD_AHL,BC   AHL:= coded address of RAMdisc segment
C5                        PUSH BC                Stack 200h * record number + 1.
D5                        PUSH DE                Stack length of record.
E5                        PUSH HL
F5                        PUSH AF                Stack page-coded address of segment.
42                        LD B,D
4B                        LD C,E                 BC:= length of record.
CD3AB7                    CALL B73A,ADD_AHL,BC   AHL:= page-coded address of byte
                                                 following RAMdisc segment.
47                        LD B,A                 BHL:= this address.
F1                        POP AF
4F                        LD C,A
D1                        POP DE                 CDE:= page-coded address of segment.
D9                        EXX                    Use alternative registers.
D1                        POP DE                 DE':= length of record.
C1                        POP BC                 BC':= 200h * record number + 1.
DDE1                      POP IX                 IX: points to channel info area.
DD7319                    LD (R_RECLEN),E
DD721A                    LD (R_RECLEN+1),D      Store length of record.
DDE5                      PUSH IX
E1                        POP HL                 HL': points to channel information.
011B00                    LD BC,001B
09                        ADD HL,BC              HL': points to
                                                 the R-channel buffer.
19                        ADD HL,DE              HL': points to byte following
                                                 current record.
0605                      LD B,05                B':= 05, signalling "standard RAM".
D9                        EXX                    Use normal registers.
DDCB198E                  RES 1,(R_CHFLAG)       Signal "Not end of file block".
08                        EX AF,AF'
D0                        RET NC                 Return unless end of file block.
DDCB18CE                  SET 1,(R_CHFLAG)       Signal "End of file block".
C9                        RET                    Return.
```

This subroutine actually assigns the R-channel buffer in preparation for use with a READ channel. Note that it calls the R_TRANSFER subroutine from label R_TRANSFER_2 in order to deal with the zero case, when the buffer is to be considered empty.

```
                          ORG B7C5
CD5DB7     R_ASSIGN       CALL B75D,R_MATCH      Match buffer with RAMdisc segment.
CD30B7                    CALL B730,R_TRANSFER_2 Copy bytes into buffer.
DD360B00   R_BUFF_EXIT    LD (R_CHBYTE),00
DD360C00                  LD (R_CHBYTE),00       Reset pointer into buffer.
C9                        RET                    Return.
```

There now follows the INPUT routine for channel R. It isolates INPUT from INKEY$ and deals with each accordingly.

```
                          ORG B7D4
2A3D5C     R_INPUT        LD HL,(ERR_SP)         HL: points to error return address.
5E                        LD E,(HL)
23                        INC HL
56                        LD D,(HL)              DE:= error return address.
217F10                    LD HL,107F,ED_ERROR
A7                        AND A
ED52                      SBC HL,DE
2021                      JR NZ,R_INKEY          Jump if dealing with INKEY$.
ED7B3D5C                  LD SP,(ERR_SP)         Clear machine stack as far as
                                                 return from EDITOR routine.
E1                        POP HL
E1                        POP HL
223D5C                    LD (ERR_SP),HL         Restore normal error return address.
CD03B8     R_INPUT_LOOP   CALL B803,R_INKEY      Input a single character into A.
FE0D                      CP 0D
C8                        RET Z                  Return if character is "enter".
FDCB377E                  BIT 7,(FLAG_X)
2007                      JR NZ,R_INPUT_2        Jump if doing INPUT LINE.
FE22                      CP 22
2003                      JR NZ,R_INPUT_2        Jump unless chr is "quotes".
CD850F                    CALL 0F85,ADD_CHAR_1   Register quotes twice.
CD850F     R_INPUT_2      CALL 0F85,ADD_CHAR_1   Insert character into INPUT area.
18EB                      JR R_INPUT_LOOP        Loop back to input rest of string.
```

The following routine inputs a single character from an R-channel and returns it in the A register.

```
                          ORG B803
CD005B     R_INKEY        CALL 5B00,SWAP         Page in ROM 0.
2A5A5B                    LD HL,(RETADDR)
E5                        PUSH HL                Stack return address in ROM 0.
D9                        EXX
C5                        PUSH BC
D5                        PUSH DE
E5                        PUSH HL                Stack alternative register set.
DD2A515C                  LD IX,(CURCHL)         IX: points to channel information.
DDCB1846                  BIT 0,(R_CHFLAG)
2804                      JR Z,R_INKEY_2         Jump if this is a READ file.
CDFCB6     R_ERROR        CALL B6FC,V_ERROR      Generate report code.
1D                        DEFB 1D                "b Wrong file type".
DD5E0B     R_INKEY_2      LD E,(R_CHBYTE)
DD560C                    LD D,(R_CHBYTE+1)      DE:= position of next byte to read.

DDCB184E                  BIT 1,(R_CHFLAG)
280F                      JR Z,R_INKEY_READ      Jump unless this is an EOF block.
DD6E19                    LD L,(R_RECLEN)
DD661A                    LD H,(R_RECLEN+1)      HL:= length of current record.
A7                        AND A
ED52                      SBC HL,DE
2004                      JR NZ,R_INKEY_READ     Jump unless we have reached the
                                                 end of the (EOF) record.
CDFCB6                    CALL B6FC,V_ERROR      Generate error report
07                        DEFB 07                "8 End of file".
DDE5       R_INKEY_READ   PUSH IX
E1                        POP HL                 HL: points to channel information.
011B00                    LD BC,001B
09                        ADD HL,BC              HL: points to buffer.
19                        ADD HL,DE              HL: points to next byte to read.
```

```
7E                      LD A,(HL)               A:= byte which INKEY$ must return.
F5                      PUSH AF
13                      INC DE                  Increment pointer.
DD730B                  LD (R_CHBYTE),E
DD720C                  LD (R_CHBYTE+1),D       Store incremented pointer.
15                      DEC D
15                      DEC D
2006                    JR NZ,R_INKEY_EXIT      Jump unless buffer to be renewed.
DD340D                  INC (R_CHREC)           Increment record number.
CDC5B7                  CALL B7C5,R_ASSIGN      Assign and reset buffer.
F1        R_INKEY_EXIT  POP AF                  A:= byte just read from buffer.
37                      SCF                     Set carry, so that INKEY$# works
                                                properly.
E1        R_INOUT_EXIT  POP HL
D1                      POP DE
C1                      POP BC
D9                      EXX                     Restore alternative registers.
E1        R_EXIT        POP HL                  HL:= return address into ROM 0.
225A5B                  LD (RETADDR),HL         Store in system variable.
C3005B                  JP 5B00,SWAP            Page in ROM 1 and return.
```

The next subroutine in designed to insert additional bytes into an already existing file stored in RAMdisc. Any files which need to be moved in order to make room for these extra bytes will be so moved, and re-indexed to accomodate. The subroutine should be entered with AHL containing the page-coded address at which to insert the bytes, and BC containing the number of bytes to insert.

```
                        ORG B85F
C5        R_MAKEROOM    PUSH BC                 Stack number of bytes to insert.
E5                      PUSH HL
F5                      PUSH AF                 Stack page-coded address at
                                                which to insert.
AF                      XOR A                   A:= 00; Carry flag reset.
67                      LD H,A
6F                      LD L,A                  AHL:= zero.
ED42                    SBC HL,BC
9F                      SBC A,A                 AHL:= minus no. of bytes to insert.
CD05B7                  CALL B705,V_SPACE       Ensure enough room for extra bytes.
F1                      POP AF
E1                      POP HL                  AHL:= address at which to insert.
C1                      POP BC                  BC:= number of bytes to insert.
C5                      PUSH BC
E5                      PUSH HL
F5                      PUSH AF
3E04                    LD A,04
CDFFB6                  CALL B6FF,V_PAGE        Select page containing catalogue.
DD2A835B                LD IX,(SF_NEXT)         IX:points to "End of cat" index.
DD6E0A                  LD L,(SF_START)
DD660B                  LD H,(SF_START+1)
DD7E0C                  LD A,(SF_START+2)       AHL:= page-coded address of first
                                                spare byte in RAMdisc area.
F5                      PUSH AF
E5                      PUSH HL                 Stack this address.
CD3AB7                  CALL B73A,ADD_AHL,BC    AHL:= page-coded address of first
                                                RAMdisc byte which will remain spare
                                                after more bytes are inserted.
47                      LD B,A                  BHL:= this address.
D9                      EXX                     BHL':= this address.
E1                      POP HL
C1                      POP BC                  BHL: pts to 1st spare byte (old).
F1                      POP AF
D1                      POP DE                  ADE:= address at which to insert.
4F                      LD C,A                  CDE:= address at which to insert.
D5                      PUSH DE
F5                      PUSH AF                 Stack this address.
CD30B7                  CALL B730,R_TRANSFER_2  Move bytes which need to be moved.
3E04                    LD A,04
CDFFB6                  CALL B6FF,V_PAGE        Select page containing catalogue.
C1                      POP BC
D1                      POP DE                  BDE:= position of insertion.
DD6E0A    R_MR_LOOP     LD L,(SF_START)         AHL:= previous (page-coded) addr of
DD660B                  LD H,(SF_START+1)       a RAMdisc file (or next spare byte)
DD7E0C                  LD A,(SF_START+2)       which may have been moved.
B8                      CP B
382E                    JR C,R_MR_FOUND         Jump if file address preceeds
                                                point of insertion.
ED52                    SBC HL,DE
19                      ADD HL,DE
3829                    JR C,R_MR_FOUND         Jump if file address preceeds
                                                point of insertion.
EB                      EX DE,HL
E3                      EX (SP),HL
EB                      EX DE,HL                DE:= number of bytes inserted.
19                      ADD HL,DE
3005                    JR NC,R_MR_ADDR
CBFC                    SET 7,H
CBF4                    SET 6,H
3C                      INC A                   AHL:= new address of file.
EB        R_MR_ADDR     EX DE,HL
E3                      EX (SP),HL
EB                      EX DE,HL                BDE:= position of insertion.
DD750A                  LD (SF_START),L
DD740B                  LD (SF_START+1),H
DD770C                  LD (SF_START+2),A       Store new start address of file.
C5                      PUSH BC
011400                  LD BC,0014
DD09                    ADD IX,BC               IX: points to index for next file.
C1                      POP BC                  BDE:= position of insertion.
DD7510                  LD (SF_END),L
DD7411                  LD (SF_END+1),H
DD7712                  LD (SF_END+2),A         Store new end address for next file.
18C6                    JR R_MR_LOOP            Loop back to deal with this file.
DD6E0D    R_MR_FOUND    LD L,(SF_LEN)
DD660E                  LD H,(SF_LEN+1)
DD7E0F                  LD A,(SF_LEN+2)         AHL:= previous length of file.
EB                      EX DE,HL
E3                      EX (SP),HL
EB                      EX DE,HL                DE:= number of bytes inserted.
19                      ADD HL,DE
CE00                    ADC A,00                AHL:= new length of file.
DD750D                  LD (SF_LEN),L
DD740E                  LD (SF_LEN+1),H
DD770F                  LD (SF_LEN+2),A         Store new length of file.
78                      LD A,B                  A:= page-code of point of insertion.
42                      LD B,D
4B                      LD C,E                  BC:= number of bytes inserted.
E1                      POP HL                  AHL:= page-coded address of point
                                                at which bytes were inserted.
C9                      RET                     Return.
```

The following subroutine will transfer the contents of the R channel buffer into the corresponding RAMdisc file.

```
                        ORG B8F2
DD4E0B    R_STORE       LD C,(R_CHBYTE)
DD460C                  LD B,(R_CHBYTE+1)       BC:= number of bytes in buffer.
DDE5                    PUSH IX                 Stack address of R channel info.
C5                      PUSH BC                 Stack number of bytes in buffer.
CD47B7                  CALL B747,FIND_FILE     IX: points to file entry in cat.
C1                      POP BC                  BC:= number of bytes in buffer.
DD6E10                  LD L,(SF_END)
DD6611                  LD H,(SF_END+1)
DD7E12                  LD A,(SF_END+2)         AHL:= page coded address of first
                                                byte beyond end of file.
CD5FB8                  CALL B85F,R_MAKEROOM    Insert enough room for contents
                                                of buffer.
DDE1                    POP IX                  IX: points to channel information.
CD3AB7                  CALL B73A,ADD_AHL,BC    AHL: points one byte beyond the
                                                last of the new bytes.
C5                      PUSH BC                 Stack number of bytes in buffer.
47                      LD B,A                  BHL:= addr of last new byte + 1.
D9                      EXX                     BHL':= addr of last new byte + 1.
DDE5                    PUSH IX
E1                      POP HL                  HL:= address of channel information.
011B00                  LD BC,001B
09                      ADD HL,BC               HL: points to start of buffer.
C1                      POP BC                  BC:= number of bytes in buffer.
E5                      PUSH HL                 Stack address of start of buffer.
09                      ADD HL,BC               HL: points to byte beyond buffer.
D1                      POP DE                  DE: points to start of buffer.
010505                  LD BC,0505              BHL and CDE now page-coded addresses.
CD30B7                  CALL B730,R_TRANSFER_2  Copy buffer into RAMdisc area.
78                      LD A,B                  A:= 05.
CDFFB6                  CALL B6FF,V_PAGE        Page in normal RAM.
C3CBB7                  JP B7CB,R_BUFF_EXIT     Reset pointer into buffer and return.
```

Now comes the output routine, whose job it is to print the character stored in the A register to an R channel (ie to store it firstly in the buffer, and ultimately in a RAMdisc file).

```
                        ORG B92B
CD005B    R_PRINT       CALL 5B00,SWAP          Page in ROM 0.
2A5A5B                  LD HL,(RETADDR)
E5                      PUSH HL                 Stack return address into ROM 0.
D9                      EXX
C5                      PUSH BC
D5                      PUSH DE
E5                      PUSH HL                 Stack alternative register set.
DD2A515C                LD IX,(CURCHL)          IX: points to channel information.
DDCB1846                BIT 0,(R_CHFLAG)
CA18B8                  JP Z,B818,R_ERROR       Error if this is a READ file.
DD5E0B                  LD E,(R_CHBYTE)
DD560C                  LD D,(R_CHBYTE+1)       DE:= number of bytes in buffer.
DDE5                    PUSH IX
E1                      POP HL                  HL: points to channel information.
011B00                  LD BC,001B
09                      ADD HL,BC               HL: points to start of buffer.
19                      ADD HL,DE               HL: points to next spare byte.
77                      LD (HL),A               Store byte in buffer.
13                      INC DE                  DE:= new no. of bytes in buffer.
DD730B                  LD (R_CHBYTE),E
DD720C                  LD (R_CHBYTE+1),D       Store new no. of chrs in buffer.
15                      DEC D
```

```
15                      DEC D
CCF2B8                  CALL Z,B8F2,R_STORE       If buffer is now full, then empty
                                                  contents into RAMdisc file.
A7                      AND A                     Reset carry.
C354B8                  JP B854,R_INOUT_EXIT      Jump to exit routine.
```

Next we have the routine to CLOSE an R channel. All that is necessary here is that the buffer contents be ignored (READ file) or stored in RAMdisc (WRITE file).

```
                        ORG B960
CD005B     R_CLOSE      CALL 5B00,SWAP            Page in ROM 0.
2A5A5B                  LD HL,(RETADDR)
E5                      PUSH HL                   Stack return address into ROM 0.
DDE5                    PUSH IX                   Stack ptr to channel information.
2A3D5C                  LD HL,(ERR_SP)            HL: pts to error return address.
E5                      PUSH HL                   Stack this pointer.
21FEFF                  LD HL,FFFE
39                      ADD HL,SP                 HL:= SP minus two.
223D5C                  LD (ERR_SP),HL            Set new error return address.
DDCB1846                BIT 0,(R_CHFLAG)
C4F2B8                  CALL NZ,B8F2,R_STORE      If this is a WRITE file, then
                                                  empty buffer contents into RAMdisc.
E1                      POP HL                    NOTE: this is also the return point
                                                  from any errors that may have
                                                  occurred during R_STORE.
223D5C                  LD (ERR_SP),HL            Restore error pointer to normal.
DDE1                    POP IX                    Restore channel info pointer.
215827     R_OC_EXIT    LD HL,2758
D9                      EXX                       HL':= 2758 to prevent crash.
C358B8                  JP B858,R_EXIT            Jump to exit routine.
```

At last we have the routine to OPEN an R channel. On entry the A register must contain the stream number to which the channel is to be attached, and the ten-byte system variable R_STR1 must contain the filename of the READ or WRITE file to be opened. If the filename is less than ten characters long then it should be followed by trailing spaces.

```
                        ORG B988
CD005B     R_OPEN       CALL 5B00,SWAP            Page in ROM 0.
2A5A5B                  LD HL,(RETADDR)
E5                      PUSH HL                   Stack return address into ROM 0.
F5                      PUSH AF                   Stack stream number.
3E52                    LD A,"R"                  A:= name of this channel ("R").
CD94B5                  CALL B594                 Search for an existing R channel.
381F                    JR C,R_OP_OK              Jump if none found.
DDE5       R_OP_LOOP    PUSH IX
E1                      POP HL                    HL: points to channel information
                                                  for already existing R channel.
010B00                  LD BC,000B
09                      ADD HL,BC                 HL: points to filename for
                                                  already existing R channel.
11675B                  LD DE,R_STR1              DE: points to intended filename
                                                  for this channel.
060A                    LD B,0A                   B:= length of filenames.
1A         R_OP_NAME    LD A,(DE)
13                      INC DE
BE                      CP (HL)
23                      INC HL
2006                    JR NZ,R_OP_RETRY          Jump if filenames are different.
10F8                    DJNZ R_OP_NAME            Test all ten characters of filename.
                                                  If filenames are identical then
CDFCB6     R_OP_ERROR   CALL B6FC,V_ERROR         Generate error report
20                      DEFB 20                   "e File already exists".
1652       R_OP_RETRY   LD D,"R"                  D:= name of this channel ("R").
CDAAB5                  CALL B5AA                 Search for next existing R channel.
30E1                    JR NC,R_OP_LOOP           Loop back if one found.
CD08B7     R_OP_OK      CALL B708,V_FIND          Search for RAMdisc file with
                                                  given name.
F5                      PUSH AF                   Stack the zero flag.
2811                    JR Z,R_OP_CONT            Jump if no file found (ie if this
                                                  is to be a WRITE file).
DD6E0A                  LD L,(SF_START)
DD660B                  LD H,(SF_START+1)
DD7E0B                  LD A,(SF_START+2)         AHL:= page-coded address of file
                                                  with given name.
CDFFB6                  CALL B6FF,V_PAGE          Select page containing first byte
                                                  of this file.
7E                      LD A,(HL)                 A:= type-of-file code.
FE04                    CP 04
20DE                    JR NZ,R_OP_ERROR          Give error unless this is a READ
                                                  file.
3E05       R_OP_CONT    LD A,05
CDFFB6                  CALL B6FF,V_PAGE          Page in normal RAM.
F1                      POP AF                    Retrieve zero flag.
08                      EX AF,AF'                 Store in A'.
F1                      POP AF                    A:= stream number.
08                      EX AF,AF'                 A':= stream number.
F5                      PUSH AF                   Stack the zero flag, which determines
                                                  whether this is a READ or WRITE file.
3E52                    LD A,"R"                  A:= name of this channel ("R").
011B02                  LD BC,021B                BC:= length of channel info block.
11D4B7                  LD DE,B7D4,R_INPUT        DE:= address of input routine.
212BB9                  LD HL,B92B,R_PRINT        HL:= address of output routine.
DD2160B9                LD IX,B960,R_CLOSE        IX:= address of close routine.
EF6DB0                  RST 28/DEFW B06D,OPEN_NEW Create channel information block.
DDE5                    PUSH IX
E1                      POP HL                    HL: points to channel information.
010B00                  LD BC,000B
09                      ADD HL,BC                 HL: points to variable R_CHBYTE.
70                      LD (HL),B
23                      INC HL
70                      LD (HL),B                 Reset R_CHBYTE.
23                      INC HL
70                      LD (HL),B                 Reset R_CHREC.
23                      INC HL
EB                      EX DE,HL                  DE: points to R_CHNAME.
21675B                  LD HL,5B67,R_STR1         HL: points to filename given.
0E0A                    LD C,0A                   BC:= length of filename (ten).
EDB0                    LDIR                      Copy filename into channel info.
F1                      POP AF                    Retrieve zero flag.
2809                    JR Z,R_OP_WRITE           Jump if this is to be a WRITE file.
DDCB1886                RES 0,(R_CHFLAG)          Signal "This is a READ file".
CDC5B7                  CALL B7C5,R_ASSIGN        Assign buffer from RAMdisc file.
1840                    JR R_OP_EXIT              Jump to exit routine.
DDCB1806   R_OP_WRITE   SET 0,(R_CHFLAG)          Signal "This is a WRITE file".
CD02B7                  CALL B702,V_NEWCAT        Create new catalogue entry.
21FFFF                  LD HL,FFFF
7C                      LD A,H                    AHL:= minus one.
CD05B7                  CALL B705,V_SPACE         Ensure enough room for one byte.
3E04                    LD A,04
CDFFB6                  CALL B6FF,V_PAGE          Select page containing catalogue.
DD6E0A                  LD L,(SF_START)
DD660B                  LD H,(SF_START+1)
DD7E0C                  LD A,(SF_START+2)         AHL:= page coded address of first
                                                  spare byte in RAMdisc.
F5                      PUSH AF                   Stack page-code.
CDFFB6                  CALL B6FF,V_PAGE          Select page containing first
                                                  spare byte in RAMdisc.
F1                      POP AF                    AHL: points to 1st spare byte.
3604                    LD (HL),04                Store 04 as type-of-file code.
010100                  LD BC,0001
CD3AB7                  CALL B73A,ADD_AHL,BC      AHL:= page-coded address of new
                                                  first spare byte in RAMdisc.
5F                      LD E,A                    EHL:= this address.
3E04                    LD A,04
CDFFB6                  CALL B6FF,V_PAGE          Select page containing catalogue.
DD7510                  LD (SF_END),L
DD7411                  LD (SF_END+1),H
DD7312                  LD (SF_END+2),E           Store address of end of file.
CD0BB7                  CALL B70B,V_CATEND        Tidy up catalogue entry.
3E05                    LD A,05
CDFFB6                  CALL B6FF,V_PAGE          Select normal RAM.
C381B9     R_OP_EXIT    JP B981,R_OC_EXIT         Jump to exit routine.
```

And finally, we have the routines which integrate the R channel with BASIC. These are only example routines, and you may of course rewrite them to your own specifications. OPEN_4 will open stream four to a serial file called FILE1; OPEN_5 will open stream five to a serial file called FILE2; CLOSE_4 will close stream four; and CLOSE_5 will close stream five.

```
                        ORG BA4D
46494C4531 FILE_1       DEFM FILE1
2020202020              DEFM five spaces          Name of first file.
46494C4532 FILE_2       DEFM FILE2
2020202020              DEFM five spaces          Name of second file.
3E04       OPEN_4       LD A,04                   A:= stream number.
214DBA                  LD HL,FILE_1              HL: points to filename.
1805                    JR OPEN_4_5               Jump to open stream.
3E05       OPEN_5       LD A,05                   A:= stream number.
2157BA                  LD HL,FILE_2              HL: points to filename.
11675B     OPEN_4_5     LD DE,R_STR1              DE: points to system variable.
010A00                  LD BC,000A                BC:= length of filename (ten).
EDB0                    LDIR                      Copy filename into system variable.
C388B9                  JP B988,R_OPEN            Jump to open channel.
3E04       CLOSE_4      LD A,04                   A:= stream number.
1802                    JR CLOSE_4_5
3E05       CLOSE_5      LD A,05                   A:= stream number.
C300B0                  JP B000,CLOSE_NEW         Jump to close channel.
```

STREAMS AND CHANNELS

WAR GAMES

Gordon Hamlett reports from the front . . .

The number of wargames available for the home computer has escalated considerably over the last couple of years. There are several reasons why the micro version of a game is succeeding whereas its traditional 'boxed game' counterpart had only limited appeal.

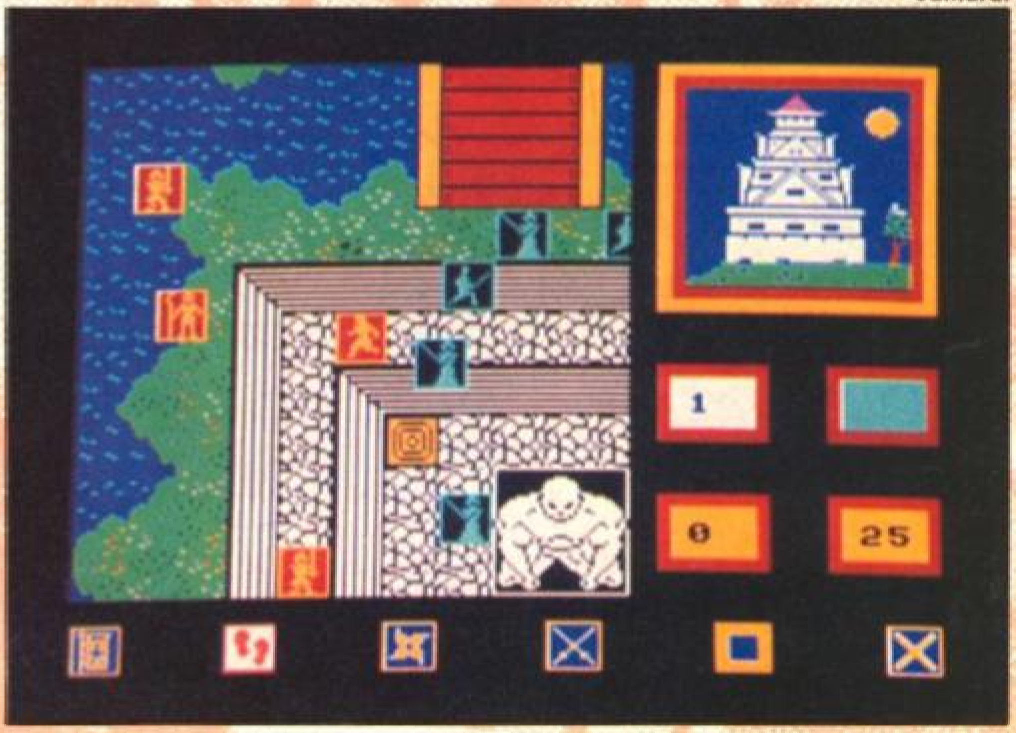
The obvious benefit that a micro offers is that you don't need a second person if you want to play a game. Then, there is no need to set up thousands of counters on a board or work out combat results from a set of complicated tables. The drudgery is all handled for you. On the debit side, computer wargames are not yet as sophisticated as their original counterparts although the balance is shifting and, as I will point out later, there are certain advantages in playability to be gained on the micro.

Three games have appeared for the Spectrum in recent weeks and all three offer the budding strategist something different. The first of these games is **Samurai** from CRL (£9.95). This is a one player game with three different scenarios and three difficulty levels.

You take the side of a small band of warriors, trying to wipe out an enemy force of assorted temple guards. The game is icon driven and starts with you selecting your initial forces. You have so many points to 'spend' and each type of warrior costs a varying amount. There are four types to choose from — Ashigari or lightly armoured troops, the traditional Japanese warrior the Samurai, a mounted Samurai and the deadly Ninja.

The mechanics of the game are very simple and are basically move and fight. Ninjas must be carefully managed as they are the only troops who can attack from a distance and should be used very much in hit-and-run tactics.

Samurai is by far the simplest of the three games and will appeal more to the novice. Beware though! To all intents and purposes, Samurai is the same game as Swords of Bane from CCS and it is probably not worth getting both.



Samurai



Battle of Britain

Moving forward a few hundred years brings us to 1940 and PSS's **Battle of Britain** (£8.95). Hitler had commanded Goering, his head of the Luftwaffe, to destroy the RAF prior to a planned invasion of Britain. Due to the RAF's meagre resources, just about any tactic would have worked provided the Germans had maintained it. In practice, Goering decided that his methods weren't working and so switched his forces to night bombing major cities. Horrendous though the blitz was, there was no way it was ever going to destroy the British planes for the simple reason that the Spitfires and Hurricanes never flew at night!

There are three main scenarios for you to try. The training game gets you used to commanding the forces at your disposal by simulating a light raid. In Blitzkrieg, the Germans throw everything they have at you but only for a period of one day. Finally, there is the much longer campaign which is played over thirty days. There are also optional arcade sequences in which you can try shooting down Messerschmidts from the comfort of your own mess room. If you choose this option, how well you do in your own personal combat directly affects the outcome of a particular battle. This is fine for arcade fans but those of you only interested in the strategy elements should leave well alone.

The gameplay depends on you making a lot of very fast decisions. As the German forces start to appear, you must scramble squadrons to intercept them. After combat or a prolonged patrol, a squadron must be landed in order that it can refuel and reload. Failure to monitor the status of your squadrons will result in them becoming dispersed and unavailable to you for a period of time. Airfields closing because of the weather add to your problems.

Again, a fairly simple game to play but as any one who has ever tried to juggle will tell you, it is very easy to throw a lot of balls up into the air at once. It is a different matter to keep them there.

The final game this month is **Vulcan** (£8.95) from CCS. This simulates the Tunisian campaign of 1942-43 and is by far the most detailed of the three games on offer. Control however remains straightforward and is all menu driven.

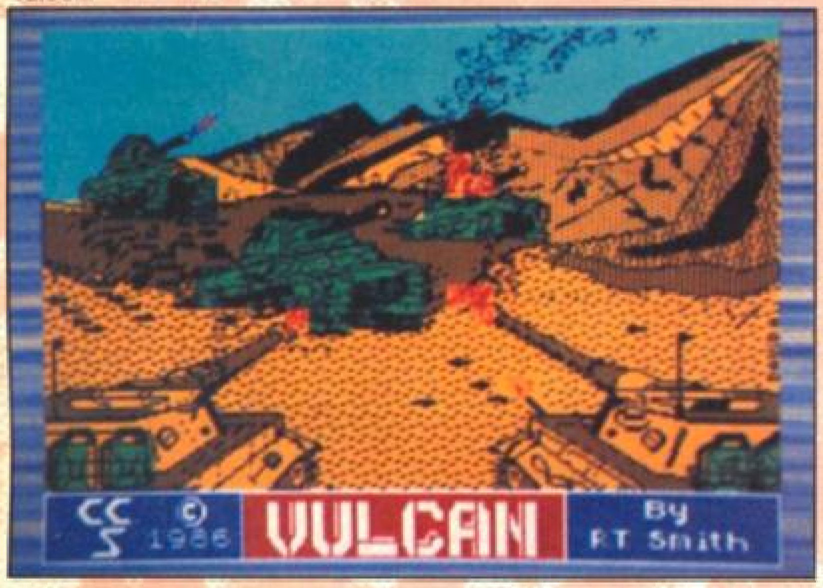
There are five different scenarios designed to last anywhere from half an hour to sixteen hours. You can choose to play either the Axis or the Allied forces and the game can be played against either a computer or human opponent. The 128 version offers several other options including a debriefing mode, several historical 'what-if' variations and no need to reload any data after every game (necessary on the 48K Spectrum due to memory restrictions).

One feature not available on board war games for obvious reasons is that of hidden movement by the enemy. Enemy units are only discovered literally when you bump into them. There are several ways of moving your troops. Normal movement, assault and travel. Assault means that a unit will go all out to gain its objective whereas travel is very defensive — you can move only along roads (at double speed) but are very vulnerable to attack. You can also choose to hold a position or fortify it.

Terrain plays an important part in the game. Not only does it affect the rate of movement, but also how well a unit can attack from or defend a position. Combat is decided by many factors including the strength of a unit, how well it is supplied and the weather. Supplying your units is a vital element in your strategy and you should also make good use of any air power that you have, either to deliver an air strike or reconnoitre as you try to discover the enemy's positions.

Vulcan is very well presented with an excellent instruction booklet giving players hints and the historical background to the campaign. Highly recommended to serious strategists.

Vulcan

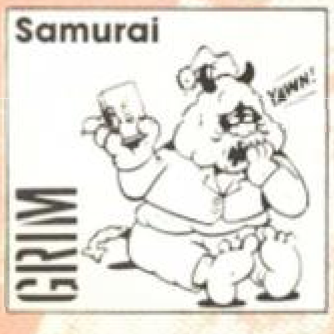


Vulcan



Battle of Britain

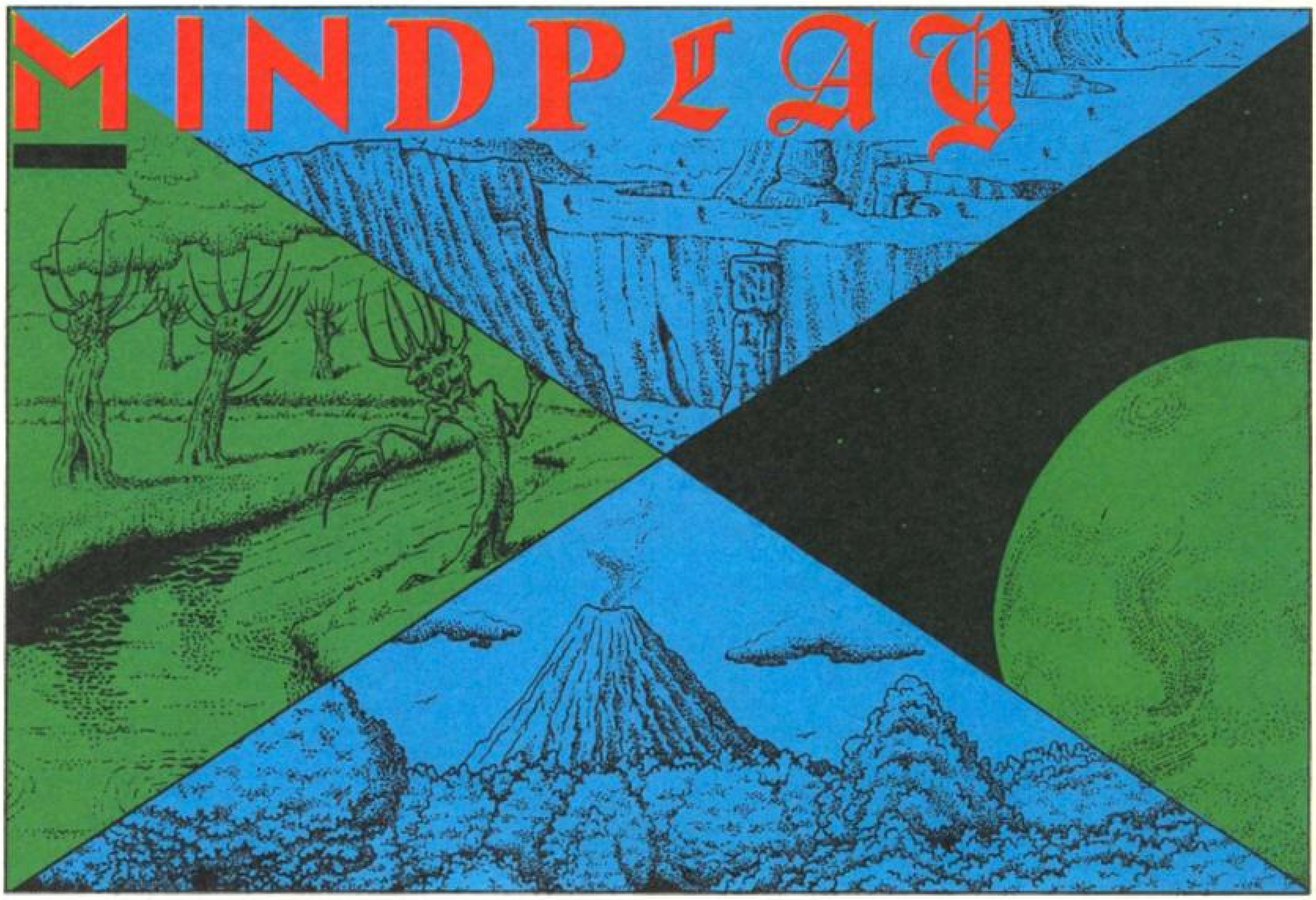








MINDPLAY



Peter Sweasey having some problems pigeon holing this month's unusual adventures.

Funny old month: none of the games for review is entirely conventional. Imagination — another cheapie from prolific author Peter Torrance — is four mini-games in one — yet is the closest to conventional.

The Growing Pains Of Adrian Mole is another Level 9 multi-choice adventure. In the past these have been bought to us from Mosaic, but this time the game comes courtesy of Richard Branson's Virgin Games — Mosaic billed as "producers". I hope this continues as a partnership; while Mosaic have been responsible for some commendably unusual licencing deals over the past few years, their advertising has been far too subtle for the games to achieve the high sales they deserved.

Third curiosity is The Inheritance from the cross-Channel software invaders at Infogrames. It is so hard to classify, I almost reviewed it elsewhere in the magazine. But while it's not straight adventure (no text input, nor much to read), nor is it really an arcade game (no finger-reactions needed). In the end, I decided categorising games is a bit futile anyway — so I reviewed it here.

Talking of France's finest, Infogrames have a very interesting product lined up for release soon. Murder On The High Seas is a complex adventure featuring 40 characters, including suspects to question, but what especially appeals to me is the packaging. Included with the game will be evidence — such as cartridge cases, hair, newspaper cuttings, ink — which you can examine when it's referred to in the game. Such packaging adds greatly to the atmosphere of an adventure, and is standard with games from Infocom (the world's leading adventure writers — unfortunately only for disc based machines), and quite common among American companies. But Spectrum adventures have virtually never seen anything like it. The closest attempt was the original release of Valkyrie 17, where you were given some documents, a stylish badge and audio messages, containing information vital to the game's completion. Runestone has a



Infogrames: providing the clues

map and short story, while Rainbird's recent adventures included short novels, but these are nowhere near as exciting as what Infocom normally offers (with their recent release Leather Goddesses of Phobos, for example, a scratch and sniff card was enclosed, for reference at certain points in the game, along with 3D glasses to read a special comic). Let's hope Infogrames set a trend . . .

Users of the Graphic Adventure Creator will be interested to learn that Incentive has published "The GAC Adventure Writers Handbook" as a supplement to the manual. It resembles a club fanzine, but contains useful information: there are clearer explanations of the program's more complicated functions, guides on how to use the program best and notes on some of its foibles.

There's also a short but excellent introduction on how to write a good adventure, covering similar points to the ones I often expound on these very pages. The handbook costs £1.25 from the Incentive address.

HELPLINE

Starting this month with RamJam's highly recommended but tricky **Terrors of Trantoss.** Both Mike Pulpher and Keith White are being foxed by the Fanglizard. Simply follow your country code: CLOSE GATE behind you. To catch the dog, Keith, you must PART the brothers. Make one take the sack to the side of the Temple, and PUT SACK INTO HOLE. Then make the other enter the temple, CLOSE DOOR and attempt to CATCH DOG. The mutt should run through the hole in the wall — straight into the sack! As for the well, SHINE LANTERN down it to see the rungs. Get Lobo to fetch the Temple door, and drop it down the well. When Scarn climbs down he will find it has broken into planks, which can be used to build a bridge across the river.

Zzzz is not the sort of noise you should make when reading this column, but it is the title of a somewhat unplayable Mastertronic cheapie. Miss L. V. Burton from Clapton cannot board the bus having passed the crocodile, even though she has the bus stop sign. A question of precision vocabulary, I suspect. ERECT SIGN, STOP BUS and GIVE MONEY.

Stephen Lawrence cannot find the magic scissors or the key in part three of **Bored Of The Rings.** ENTER FORGES to find the latter, while the former are found in the bar, W from the headless statue (you'll have to kill the Kremlins first).

Dead End Quest

Neil Talbott is flummoxed by Melbourne House's rather good **Mordon's Quest** (or is it Mastertronic's now?). In the deserted beach he wants to stay underwater. Head out and CLIMB INTO boat, go down and, as you suspected, you'll find an aqualung. Then, to prevent suffocation, N,N,N,SE,U,SE,D,N,E and FILL AQUALUNG. Progress in this sector will find you the spray paint you need in the future: you'll also find someone who wants that newspaper you asked about (though how it doesn't go all soggy is unexplained). The keypad combination, incidentally, is 8875. There's no need to unlock the doors in the Roman villa, indeed there's no need to enter it, at least not to complete the game.

Andrew McMaster wishes to pass the television camera in **Seabase Delta.** You need to throw a pancake at it! How do you make a pancake, you cry? You'll need the hen's egg (blow a chewing gum bubble to wake the poultry), and the bowl, which is on the conveyor belt if I remember rightly. Examine the kitchen shelf and take the flour from it, then open the fridge to find the milk. MAKE PANCAKE, COOK PANCAKE, GET PANCAKE (how you manage to carry it around without it falling to bits or sticking to your hands is not explained — realism is not a strong point in this game).

Marcus Beer wishes to kill Stripe in that anarchic adventure, **Gremlins,** having already blown up the store. As the gas exploded, the not-so-cute one should have rushed away and knocked you over. So FOLLOW STRIPE and he'll lead you to the swimming pool. You should have drained this earlier, so when Stripe dived in to reproduce he will have knocked himself out. GET the unconscious fiend, head UP then go east. Dawn will break — and you know how gremlins hate light . . . To rescue Antman in The Hulk, first PLUG EARS, HOLD NOSE and CLOSE EYES. Then GET ANTS, north twice to the underground room, go through the crack and they will attack Ultron. As for **Valkyrie 17,** if you go to the Carpathians in the taxi, you won't be able to pass the guard. However, where to really go in the taxi, is answered by solving the other problem Marcus asks about — how to use the telescope. You need to INSERT RINGPULL in the slot, then TURN TELESCOPE for your destination. (The ringpull comes from the can in the fountain — kill the fish with the liquid).

Gremlin Gripes

Help for Simon Fuller with the same game: to steal the lamb, just take it, then give the butcher that small lead box from the bedroom safe (you know, the one with the glowing stuff in it). Simon also asks for help with about a squillion other adventures, so here we go. In Project X, The Microman he cannot escape the car: CLIMB SEAT BELT, WIND HANDLE and OUT (take the lighter first). In **Dracula Part Two** (I hope you're over 15, Simon) progress from the first location is obtained like this: LOOK AROUND, LOOK OLD WOMAN and LOOK EYES. Take what she offers you, but hide it. In The Final Mission, Simon wants to avoid the slime. In fact, dealing with the green stuff will solve you another problem if you prepare properly. Take the straw (from the northern corner of the large cell) and use it to fill the one inch gap at the bottom of the door. Making sure you have the soap (from south of the great pillared hall), go to the chamber with the rising, damp breeze, and head up, into the cubby hole. Wait until you hear a splintering sound, then DROP SOAP. Not only will the slime have been cleaned up, but it will have smashed the door for you! As for the native woman in **Espionage Island** (a game banned by W.H. Smith for one of its responses!), you must give her the beads.

Simon asks many more questions, but space is running out, so we'll save those for next time.

Peter Sweasey for Poet Laureate

"If you're stuck in a game
And your hair's all torn out
Then remember my name;
Just give me a shout!
Fill in the coupon
It's as easy as pie
Send it to Mindplay (well you try thinking of a rhyme for 'coupon')
And to solve I will try."

(Er, I think you better stick to reviewing, Sweasey — Ed). Philistine! A few rules: British correspondents, please enclose a stamped, addressed envelope if you want a personal reply rather than wait some months for

the magazine to come out. If you are writing from abroad, just enclose an envelope — I'll add the postage. I try to respond within two months but I can take longer (on the other hand, you might receive an immediate reply). I ONLY DEAL WITH ADVENTURES. Not arcade games; nor technical problems (write to Crosswires about those); nor arcade adventures (Gargoyle games included, not even Heavy on The Magick). Please put the name of the game you're writing about on the back of the envelope. And PLEASE don't write letters asking for general solutions — I just don't have time to write these out for everyone. (They can be obtained, however, from good adventure clubs). Finally, I don't advise you write to me with Level 9 problems. Their hintsheets are really much better than anything I could offer, so any Level 9 questions come way down my list of priorities.

The address for problems, critical appreciations and offers for publishing rights is: Mindplay, ZX Computing, No 1 Golden Square, London W1R 3AB. How many other magazines solve your problems like this AND keep you culturally entertained?

Title: ....

Company: ....

Problem: ....

....

....

I can help solve: ....

Name: ....

Address: ....

....

THE GROWING PAINS OF ADRIAN MOLE

Virgin
£9.95

Like the first Mole game (published some time ago), this has been programmed by Level 9 and is in a very similar style (the one also used for The Archers). Just in case you've been spending the past few years with your head in a bucket, Adrian Mole is an angst-ridden teenager who thinks he's a trendy intellectual (in fact he's naive and, frankly, square), has squabbling, unemployed parents and loves precocious Pandora. The phenomenally successful books (which spawned high-rating television series) were in diary format (which the game retains). The Growing Pains sees Adrian with 'O' Levels approaching, and is set around 1982-3, the time of the Falklands war.

The game is in four loads, each dealing with several months of Ade's life. Excerpts from his diary scroll up (there are some graphics, but they're well down to Level 9's abominable standard, abstract beyond belief and best turned off) until, after certain situations, you must choose one of three options as to what Adrian can do. This may well have the effect of causing more options to appear, as a whole sub-plot is pursued. The aim is to make Adrian as popular as possible (there's a percentage score and a rating); or you could try to do the opposite!



I don't think you'll understand this game if you know nothing about Adrian Mole, as there's no explanation in the instructions about who the characters you'll encounter are. You don't have to be a Mole fanatic to appreciate the game though; I'm certainly not, yet I enjoyed it.

I think fans of the book will enjoy the game due to the chance to make Adrian act differently from the way he does on page. However, this means programmer Pete Austin has had to add totally new situations (the results of different choices); unfortunately some of these are not at all in the style of the book. What really annoyed me was his total disregard for the book's scrupulous period setting (it is a diary, after all). I wasn't very far from Adrian's age in 1982, so I was quite nostalgic at Sue Townsend's references to events such as the start of breakfast television (setting the alarm extra early!). Then it's ruined with a reference to Dirty Den (EastEnders began in 1985 for goodness sake!); or giving the option to see Jaws or Superman at the cinema (both were films of the seventies that wouldn't have been playing in 1982). Mr Austin should have done what all good authors do — research what he's writing about. I was also puzzled at some of the changes he made to Sue Townsend's original text. While tight memory obviously prevents the computer version being identical, in some instances Pete Austin has changed little details — like who said what, or a person's reactions — yet not made the diary entry noticeably shorter. Why? What gives Mr Austin the right to tamper with what many feel is a modern classic?

As I just hinted, the major problem with the transition from book to game is that due to memory restrictions, the text is severely pruned. This means many of the little observations are lost, along with much of the detail (such as letters); yet it was this detail which I liked so much with the book. You could read the book while playing the game, but that seems a pointless exercise. I can't help feeling this project should never have been attempted on cassette — disc should have been used (even if a Spectrum version was thus made impossible), and the complete book transferred.

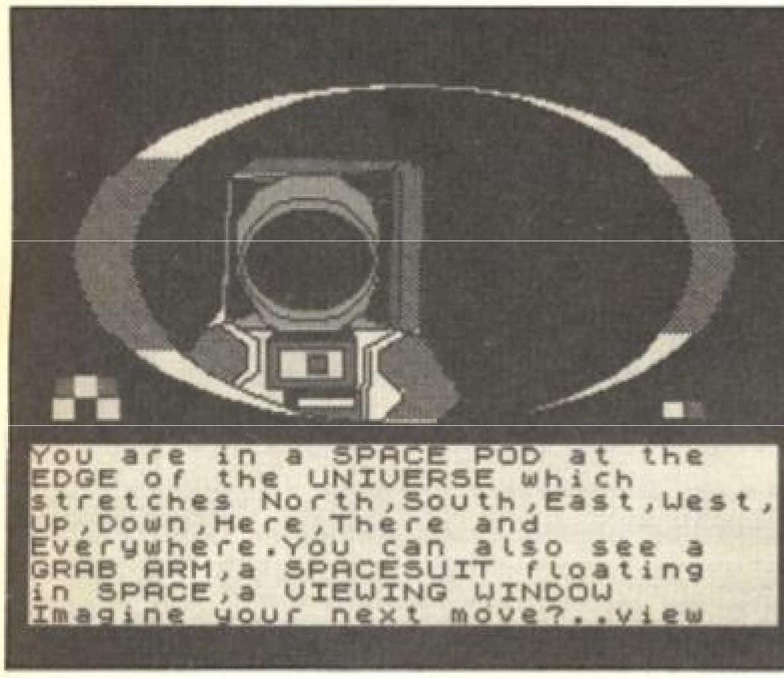
One of the problems with The Archers game was contradicting storylines — a character you had sent away would immediately reappear. This doesn't happen with Adrian Mole, mostly because few of the decisions are of long term importance. If this game was more sophisticated, one decision could utterly alter the rest of the adventure, parts of the plot being made completely unfeasible. Again, a possibility if disc had been used.

As with The Archers, Growing Pains has questionable durability, since after playing several times you'll have seen all it has to offer. The game is a fun extension of the books, and should offer some entertainment. But I feel that because it was impossible to do full justice to the original text (ie print it all), the project should not have been attempted.



IMAGINATION

Firebird
£1.99



With a title like that, Firebird are setting themselves up for criticism. Indeed, I was going to start this review by telling you how utterly devoid this product is of any original and creative thought — but then I discovered an endearing and unique feature, which made the title a little more justified than I first thought.

The game starts with you sitting in front of your computer; you insert an unlabelled disc (somewhat unlikely with the Spectrum but never mind) purchased from a bargain box and are presented with a choice of four games. Choosing one of these — a space adventure, war game ("Raid Over Margate"), a fantasy adventure ("Lords of Half Past Nine") and a platform arcade game — draws you into the screen. (A recap for the confused: this is four mini-adventures in one). Pinching your arm brings you back to reality.

In an age where single games are using several loads in order to provide more gameplay, several games in a single load seemed a strange idea. Necessity of memory means each has only a handful of locations. What prevents me from utterly slating Imagination is that you are not supposed to play each game as an individual adventure, rather move constantly between them, taking objects from one 'game' to another. This gives the problem solving an engaging new angle.

Otherwise, Imagination is unsatisfactory. It's WRITTEN by Peter TORRANCE (Seabase Delta etc) which means the return OF those IRRITATINGLY random CAPITAL letters in the TEXT. It bears all the other hallmarks of his adventures, most notably a sort of childish feel, as if were written by a twelve year old: simplistic text, laughably unsubtle clues, generally weak humour. But Imagination also features the other Torrance trademark; entertaining and sometimes ingenious puzzles. Then again, this plus point may be outweighed for you by a number of further faults. The vocabulary seems painfully small (and there are no single letter abbreviations for LOOK or INVENTORY — virtually inexcusable), and what you can do is limited — largely, only those actions necessary to complete the game emote a response, which I've said countless times before is poor adventuring. But not only are you unable to do much, the messages which tell you so are aggravating and long winded, along the lines of "Well you can IMAGINE that . . . but you can't do it."

Like Torrance's Apache Gold, reviewed last month, this is GACed — and I still think Quilled games are far superior. The graphics are mostly dull and do not enhance the game in the least. Of course they might seem better if the presentation was not so dire. Glaring white background with black, normal Spectrum lettering (so the CAPITALS stand out even more), and sloppy looking input lines. GACed games also have more sluggish response times and scrolling.

Imagination is cheap and fun to solve — but it's truly primitive. Surely Spectrum adventures can be more sophisticated than this — even at budget price?



THE INHERITANCE

Infogrames
£9.95

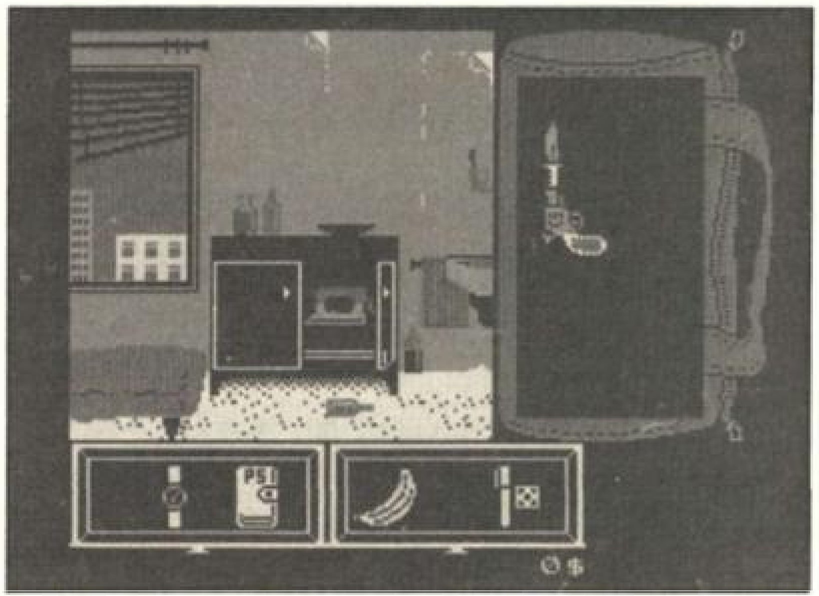
Things are looking bleak for penniless Peter when suddenly, a letter arrives; his aunt has made him sole heir to her vast fortune. However, to gain it, he must make a million dollars in Las Vegas in a single night. In game one of this three-parter, you must escape your apartment block, having first satisfied the numerous fellow inhabitants who want the return of items you borrowed from them. Part two is set at the airport, while part three is in Las Vegas itself, where you can participate in several gambling games (fruit machines, craps, etc) in your quest for the magic million.

Control is via a roving (and sometimes flickering) cursor. With this you can pick up objects, open doors and move. This gives an endearing sense of real exploration. The graphics are large, colourful and generally rather good. You encounter various characters, all well drawn in cartoon style, who utter stilted dialogue (the translation from French shows, unfortunately); their mouths are animated — impressively — to show this. There's very little text except their captions.

The Inheritance has many genuinely original touches, and is cleverly designed. Unfortunately, the gameplay is weak. The characters' mouths take ages to recite speeches you've already read countless times before, so you become bored. The tasks are mostly repetitive and tedious — especially making the money in the final part (the gambling machines are entirely based on luck — hardly enthralling tests of skill). At some points you are required to sit there and do nothing for over a minute. The game system (ie the roving cursor) has great potential — but sadly The Inheritance doesn't use it. Nice try.







ZX COMPUTING

FULL CREDIT FACILITIES FOR ACCESS AND BARCLAYCARD.

Lineage: 48p per word. (VAT inclusive)

Semi display: £9.50 per single column centimetre + VAT. Ring for information on series bookings/ discounts (minimum 2.5cm).

All advertisements in this section must be prepaid.
Advertisements are accepted subject to the terms and conditions printed on the advertisement rate card (available on request).

01 - 437 0699

Send your requirements to:
NICOLA BATY
ASP Ltd., 1 Golden Square, London W1

UTILITIES

D.D.S. THE PROGRAMMING SPECIALISTS

BASIC +
At last the power and speed of machine code with the ease of BASIC By adding over 40 new commands to your normal sinclair basic you can now easily create spectacular games and special effects with very little effort. Upto 16 different WINDOWS can be created at any position and any size on the screen, from a single character square to the whole screen.These windows can then be FILLED with any colour of ink or paper,it can be SCROLLED by attributes, pixel by pixel or by character square in any direction, you can also WRAP it so anything going off one side of the window appears on the opposite side, you can also make the contents of the windows slowly FADE away, or clear or fill it with any CHRs or UDG' the possibilities are endless and the commands so easy to use.To scroll downwards window number 5 then you simply type.. !SCROLL d 5 ...and it is done in a split second. Then their are the 16 multicoloured Hi - resolution animated sprites which are four times the normal udg sizes.These can be printed to any part of the screen, and can move at 31 different speeds, in set patterns, continiously or only whenyou program them to move. You can define the window in which they move and wheather they will wrap around or bounce if they hit the edges.You can also tell it to explode, continue or stop if it collides with something. And once programmed it won't need altering, it will erase the old sprite, find the new position, animate, draw the new sprite, detect if hit, and do whatever neccesary automatically and with unbelieveable speed. There are also many other commands which will improve the sound or inhance the graphics such as SOUND and WHITENOISE' a double poke and peek, a FILL command, a MOVE command to move blocks of code or fill areas of RAM with a set number, you can disable BREAK or jump to any line number if an ERROR occurs.There is also a built in TOOLKIT with RENUMBER,DELETE, REMOVE, DISPLAY blocks of memory(inHEX,DEC and You can also add your own commands and routines to run along with BASIC +. Wheather you are writing games, adventures or utilities BASIC + will radically increase the quality, speed and presentation of any program.All the commands are extreamly simple to use, and are typed in with the normal BASIC program lines. The program is compatable with interface 1, Microdrives and Kempston Joystick Interfaces.

SPRITE DESIGNER
The most advanced on the market, and the easiest to use. After easily drawing your sprite using a 8X magnification you can mirror, rotate, invert, scroll, clear, save and recall instantly, it even shows you the sprites as they are normal size and animated.

PRINTER
This allows you to dump any part or all the screen to a Sinclair or Epson compatable printer, the screen dump can be normal, inverted,or shaded with a different grey scale for each colour,including all the BRIGHT colours.

THE COST:BASIC + £10
the other programs £6 Each
OR
THE PROGRAMMERS PACK
All four programs on two cassettes in a large video box
Half Price ONLY £14.00

64 COLUMN DISPLAY
This program lets you print up to 64 charaters on each and every line, including the bottom two lines using the normal PRINT commands, AT and TAB. It will print single letters, strings , graphics and CHRs, you can even LIST in 64 charaters, You can mix normal 32 column and 64 column text you can even have UDGS. Ideal for information pages wordprocessors, databases, accounts programs etc. It can be used with or seperately from BASIC+

All our programs are ready and posted within 24 hours. If you have any queries phone: 021 550 6564
Please make cheques or postal orders to: D.D.S., 48 Beaumont Road, Halesowen, West Midlands B62 9HD or pay TRANS CASH No. 43 238 0701.

NEW TAPE UTILITY 9

This program gives a tape back-up of most software. Even converts most jerky loaders to normal for reliable loading. Manages very long programs (even 51k!), fast loaders, etc, etc. State type of Spectrum when ordering (48k, 128k etc). **£6.99 on tape (£4.99 if updating)**

NEW TRANSFER PACK 3 – FOR TAPE TO DRIVE

A software system to convert many TAPE based programs to your DRIVE (no matter what type) + OTHER UTILITIES. Most jerky programs are EASILY converted to drive. Pack has at least 7 programs! Opus/Beta owners add £1 for OPTIONAL extra program. **State TYPE of Spectrum + name of DRIVE when ordering** (eg 128k and Microdrive). **£11.99 on TAPE OR £12.99 on m/drive cartridge.** (£6 if updating).
INFORMATION SHEETS – £1 each – approx 8 transfers of popular programs per sheet – needs TP3. Up to No 25 available. **SPECIAL OFFER – No's 1-20 only £4.50.**
MICROTRANS – for m/drive to m/drive back-up + tape to drive (inc headerless) – does not convert programs. Includes MICROTAPE and RUN programs. ONLY £3 on TAPE. £4.50 on Cart. For any Spectrum.
CODE COMPRESSOR – £3.50
Overseas: add £1 Europe, £2 others each product. SAE for details.
NO RISK! – ALL PROGRAMS CARRY OUR MONEY BACK GUARANTEE (not updates)

LERM, DEPT ZX2, 11 BEACONSFIELD CLOSE, WHITLEY BAY NE25 9UW (Tel: 091-2533615)

PRINTER SERVICES

FABRIC RIBBON RE-INKING
'Introductory Offer'
£1.50
Send ribbons + prepayment to:
ALADDINK, (Dept ZXC), 4 Hurkur Crescent, Eyemouth, Berwickshire, TD14 5AP.
Tel: 08907 50965
See 'NEWS' ZX April 1987.

ACCESSORIES

MAKE SOME SPECTRUM CeNCe
Simple Computer Controller for Machinery and Models using your Sinclair Spectrum.
A little black box which plugs into the expansion socket of any Spectrum and enables you to drive two small to medium stepper motors (up to 2 AMPs per phase) or up to 8 DC motors, relays, lamps or any other DC device.
Simple to Program and built by an engineering firm for its own use. You can use it for CNC turning, milling or coordinate drilling.
Reliable Prototype still running after three years use.
We provide Interface instructions, software, hints and examples. Price £125 inc. carriage and VAT Power supplies and Stepper Motors available. We will be pleased to quote for larger units.
P.R. PRODUCTS, PARK MILL, HOBSON STREET, MACCLESFIELD, CHESHIRE SK11 6BE.
Tel: (0625) 22800.

UTILITIES

BRADWAY SOFTWARE (ZX)

LETTA-HEAD PLUS The most versatile Spectrum utility program for designing and printing business and personal stationery, receipts, letterheads, labels, posters, etc. Create the design on screen using inbuilt graphics or load any SCREENS from an arts package, add text from the library of 25 different fonts, select the required output format and print all the copies you need. **Price only £9.00**

DUMPY 3 Fast becoming the de facto standard screendump program, producing all the screen dumps you will ever need for your Spectrum. No need to understand assembler: **DUMPY's** code **GENERATOR** creates the screendump, relocates it, and saves it ready for use in your programs. Over 1000 different size, shape and density screendumps, anywhere in RAM! Price less than 1p per dump!! **£8.50.**

LIN-O-TYPE Add style to your written word. Print out on any wordprocessor file on tape, microdrive or disc in high denstiy NLQ in a choice of 25 fonts, or turn your Spectrum into a full WYSIWYG electronic typewriter, superb for addressing envelopes, filling in forms or writing short notes. **Price £8.50.**

ASTRUM + NEW!! The very best Z80 assembler and monitor package for the Z80 assembler and monitior package for the Spectrum: Exceedingly versatile, powerful and easy to use; Full wordprocessor style editor; Unlimited source code; Assembly using microdrives or the Opus Discovery ; Command driven monitor with multiple breakpoints, single stepping, slow run etc. **Price £13.50** (state microdrive or Opus version).

DISC MANAGER NEWEST!! A brand new utility for users of the Discovery drives: Full disc sector editor and file mangaement program: Erase, rename, copy goups of files; Examine, modify and understand disc directory information; Examine, modify and recover damaged date. **Price £10.50** including 3½" disc only.

Bradway Software programs drive most (currently 16 different) interfaces: **Letta-Head, Dumpy & Lin-O-Type** require an Epson compatible printer and are compatible with Microdrive, Wafadrive & Disc. Prices are for cassettes, include full instruction and P&P within Europe; please add £2.00 per program worldwide airmail postage, £1.50 for microdrive cartridge or disc.
Send for further details!

33 Conalan Avenue, Sheffield S17 4PG

WANTED

WANTED: YOUR non working Commodore Computers and equipment for cash. Ring D.M.R. Electronics, Bradford, (0274) 882892.

RING 01-437 0699 FOR DETAILS OF OUR SERIES DISCOUNTS

RACING SOFTWARE

TO ADVERTISE IN OUR NEWEST SECTION RING NICOLA BATY ON 01-437 0699

SERVICES

STOP PLAYING GAMES

Use your computer to make money. Turn your hobby into a home-based income. Full and part time opportunities to cash in on this tremendous market. High earnings easily possible. Open to any amateur micro user and gamer. Write for free details.

Westlink Promotions (ZX)
108 George Street
Edinburgh EH2 4LH.

WARNING NOTICE

Advertisements placed in this magazine are to be in strict compliance with our standard conditions (copies of which conditions are available on request) and on the clear understanding that the advertiser warrants that his advertisement(s) does not infringe any copyright or condition of sale of any interested party in the advertised product.

Further, the advertiser indemnifies the proprietors of this magazine in respect of costs, damages, or any other claims brought against them as a result of legal action arising from the publication of the advertisement.

Any breach of these terms or the said conditions may result in prosecution of the advertiser by the proprietors.

SOFTWARE

HUMPHREY SOFTWARE
TWO PROGRAMS

SUPER HANDICAP; The connoisseur racing program that will pay for itself. Full instructions and recommendations.
ONLY £9.95.

SEEK & FIND; Try to sink your opponents ships before he gets yours, but watch out for the mines! One or two players, many features.
ONLY £3.99

SPECIAL OFFER
BOTH PROGRAMS AT ONLY £11.95 INCL.

HUMPHREY SOFTWARE
8 Richard Andersen Court,
Monson Road, London SE14 5EJ.

CHEZRON SOFTWARE
(Dept. 303) 605 Loughborough Road
Birstall LEICESTER LE4 4NJ

BONABANNER — Programmers! Unleash the power of string graphics on your Spectrum. Create title pages, text adventure graphics, program headers etc. that print instantly from Basic or machine code.

At just £3.50 why hesitate! — if our demo program doesn't convince that **BONABANNER** is useful to you. JUST RETURN THE WHOLE PACKAGE FOR A FULL REFUND!!

QL/SPECTRUM UTILITIES! Wide range of business/practical programs available. SAE/IRC for details (state Micro). S.D. Microsystems (ZX) PO Box 24, Hitchin, Herts.

WD Software

FOR THE QL:

JOSS — **base £13**

Forget that tedious, time-consuming syntax! Just move the cursor and press SPACE for all your file commands. Cursor keys or your joystick allow you to access microdrives (up to 8) and floppy discs (as many as our interface allows) with up to 150 files on each! Scroll and print directories, COPY, DELETE or PRINT any file, select TV or Monitor mode before LOADing or RUNning any program. You only use the keyboard to set the DATE or label a device when FORMATting. Easy to use with Psion and other software. No silly icons to learn - JOSS will TELL you what it's going to do! Programmer's toolkit and mass copying/printing utilities also supplied. Specify microdrive-only, Microperipheral or CST-compatible disc versions.

Ref QL7 — **base £7**

1300 useful QL references with ARCHIVE 2 search/print program Too long for just one cartridge, so if you have RefQL5 just pay 2 and extra media cost to update.

Mdv Extension Cable — **£5.50**

Eight inches long, allows addition of extra microdrives to your QL. Twist it to put their slots facing you.

FOR THE QL: SPECTRUM; BBC; ELECTRON

WD 'orse Tutor — **base £4**

Written to teach amateurs, now used by professionals too! Absolute beginner, or stretching your speed to 18 wpm, you won't find anything with more helpful features. What else can offer 100 random sentences as well as all the basics? Disc version unsuitable for BBC B+.

FOR THE SPECTRUM:

WorDfinder (Microdrive/disc only) — **base £8**

For CHEATING at crosswords. Finds m-ss-ng letters, solves, anagrms. 13,000 word vocabulary, so › long to share a cartridge. 10-letter word ending in ATE? No problem.

Tradewind — **base £3**

Sailing/trading strategy game with graphic surprises.

Jersey Quest — **base £3**

Text adventure in time. Background of Jersey folklore from Stone Age to Bergerac.

For export:

QL hardware and software from many sources. Ask for list/quotation.

ORDERING ADD COST OF MEDIUM, POSTAGE £1 OUTSIDE EUROPE.
Mdv or 5.25" floppy = £2 3.5" floppy = £4 Cassette = £0

Payment

By ACCESS/Eurocard/MasterCard or STERLING (UK bank cheques, Eurocheques, drafts or International GIRO). To:

WD Software (ZX), Hilltop, St. Mary, Jersery, C.I. tel: (0534) 81392

FORTRAN 77S COMPILER for 48K Spectrum for just £15. Send cheque to Mira Software, 24 Home Close, Kibworth, Leics LE8 0JT. Or write for further information.

TERMS & CONDITIONS

CLASSIFIED ADVERTISING TERMS & CONDITIONS

Our terms for new advertisers (semi-display and lineage) are strictly pro-forma payments until satisfactory reference can be taken up (excluding recognised advertising agencies). Cheques and P.O.'s should be crossed and made payable to ARGUS SPECIALIST PUBLICATIONS LTD. and sent together with the advertisement to:

"The Classified Dept.,
No. 1 Golden Square,
London W1R 3AB.

There are no reimbursements for cancellations. Advertisements arriving too late for a particular issue will be inserted in the following issue unless accompanied by instructions to the contrary. It is the responsibility of the advertiser to ensure that the first insertion of every series is published correctly, and corrections must be notified in time for the second insertion, otherwise the publishers will not accept any liability or offer any reduction in charges.

All advertising sales are subject to Government Regulations concerning VAT. Advertisers are responsible for complying with the various legal requirements in force eg: The Trade Description Act, sex discrimination act & the business advertisements (disclosure) order 1977.

Full Terms & Conditions of Advertising available on request.

Lineage rate: 48p per word (VAT inc.) Minimum **£7.20.**
Semi display: £9.50 per single column centimetre + VAT. Minimum size 2cm. No reimbursements for cancellations.
All ads must be pre-paid.
Write your advert in BLOCK CAPITALS in the grid below, ticking the section you wish it to appear under, INCLUDING YOUR NAME AND ADDRESS IN THE WORD COUNT and send it to: **ZX COMPUTING, ADVERTISEMENT DEPARTMENT, NO: 1 GOLDEN SQUARE, LONDON W1R 3AB.**

☐ REPAIRS ☐ SOFTWARE ☐ UTILITIES ☐ FOR SALE ☐ ACCESSORIES ☐ OTHERWISE PLEASE STATE

CLASSIFIED COUPON

ALL CLASSIFIED ADVERTISEMENTS MUST BE PRE-PAID.
THERE ARE NO REIMBURSEMENTS FOR CANCELLATIONS.

I enclose my Cheque/Postal Order for £.........................for......
insertions, made payable to Argus Specialist Publications.
(Delete as necessary)

PLEASE DEBIT MY ACCESS/BARCLAYCARD NO

☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

£ .... FOR .... INSERTIONS **EXPIRY DATE** ....

Name ....

Address ....

.... POST CODE ....

DAYTIME TEL NO. ....

Signature .... Date ....

IF YOU DO NOT WISH TO CUT YOUR MAGAZINE, PHOTOCOPY THIS FORM

TAU CETI

When you've crammed a whole planet into 48K what more can you do for the 128 version?

Tau Ceti (The Special Edition)
CRL
£9.95
128/+2 only

Owners of the 128 and +2 have waited in vain for the torrent of games that were going to take advantage of all that extra memory. Games for the 128 only have been few and far between and "enhanced" versions of 48K games have often been disappointing, applying cosmetic graphic touches but not really exploiting the memory increase.

With Tau Ceti, a game that seemed to cram an enormous amount into 48K, the potential was obviously there to produce an upgraded version appropriate to the wide open spaces of the 128's memory. It's pleasing to report that CRL have done a good job in upgrading the original with a bigger game-play area on the planet surface, improved graphics, more atmospheric pictures of building interiors and a lot more information available at sites like the library and time vault.

For those unfamiliar with the 48K version, Tau Ceti is a sophisticated shoot-em-up of great complexity calling for a degree of strategy along with some determined zapping. The plot centres on Gal Corps efforts to recolonise Tau Ceti III. You as pilot of a "skimmer" spacecraft must shut down the fusion reactor in the capital city Centralis. Pitted against you are all manner of robot controlled defence systems.

The action takes place on and above the planet surface. If you can successfully dock with a building you can gain access to valuable information; flying your craft through an airlock to reach a docking bay requires a steady hand on the joystick.

Tau Ceti requires dedication to be enjoyed to the full, but the shoot-em-up element supplies enough instant gratification to pull you into the game proper. If you like games that you can become thoroughly absorbed in, Tau Ceti fits the bill and if you are a new Spectrum 128 owner it's a must. Whether the enhancements make it worthwhile buying a 128 version if you already have the 48K game is debatable and depends mainly on how enthralled you were by the original. Tau Ceti, The Special Edition, more than justifies its £9.95 price tag and gives a foretaste of what this virtually unexplored machine may be able to do when pushed to its limits.







THE OFFICIAL

FA Cup 87

FOOTBALL

This new up-dated version of last season's best selling official FA Cup Football incorporates every league club's form for the 1986 season as well as new manager's questions to test your judgement.

FA Cup Football 87 is a text based simulation of the FA Competition that allows you to take 10 teams to success. You make decisions about tactics during play and answer which contribute to your teams victories.

The game has been developed with the full endorsement of the Football Association and the form that gives such realistic results has been provided by sports writer Tony Williams (author of the Official FA League and Non-League Directories).

Available for Spectrum 48/128, Commodore 64/128, and Amstrad CPC computers. All versions are on cassette only and cost £7.95. Available from all good software shops from 24 February 1987 or directly from Virgin Games.



Spectrum 48/128	£7.95	☐
Commodore 64/128	£7.95	☐
Amstrad CPC	£7.95	☐

Please tick the version you require and send this tear-off to: **Virgin Games Mail Order, 2/4 Vernon Yard, Portobello Road, London W11 2DX.**

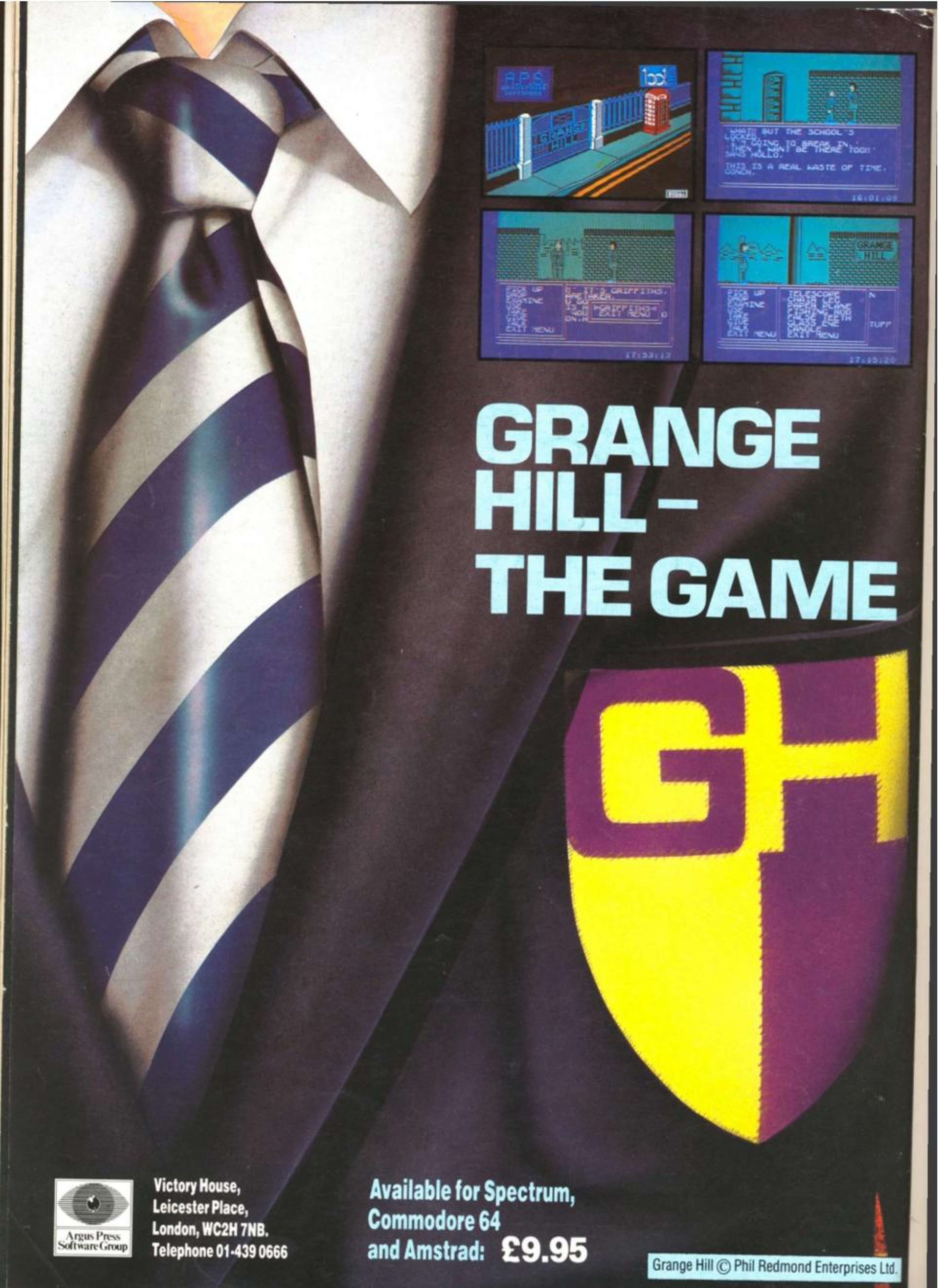
Make cheques or crossed postal orders payable to **Virgin Games Limited.** Please do not post notes or coins.

NAME ....

ADDRESS ....

....

MONEY ENCLOSED ....


GRANGE HILL – THE GAME
GH
Argus Press Software Group
Victory House, Leicester Place, London, WC2H 7NB. Telephone 01-439 0666
Available for Spectrum, Commodore 64 and Amstrad: £9.95
Grange Hill © Phil Redmond Enterprises Ltd.